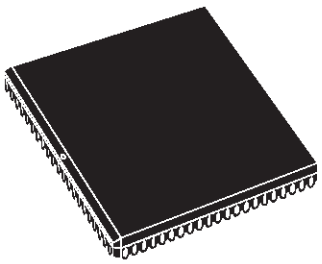
ST90158 - ST90135

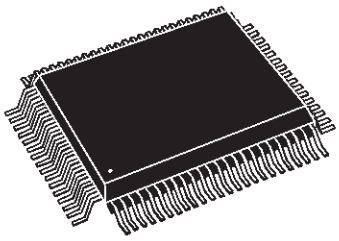
8/16-BIT MCU FAMILY WITH UP TO 64K ROM/OTP/EPROM AND UP TO 2K RAM

PRELIMINARY DATA

- Register File based 8/16 bit Core Architecture with RUN, WFI, SLOW and HALT modes
- 0 - 16 MHz Operation @ 5V±10%, 0 - 12 MHz Operation @ 3V±10% (ROM only)
- -40°C to +85°C and 0°C to +70°C Operating Temperature Ranges
- Fully Programmable PLL Clock Generator, with Frequency Multiplication and low frequency, low cost external crystal
- Minimum 8-bit Instruction Cycle time: 250ns - (@ 16 MHz internal clock frequency)
- Minimum 16-bit Instruction Cycle time: 375ns - (@ 16 MHz internal clock frequency)
- Internal Memory:
  - EPROM/OTP/ROM 16/24/32/48/64K bytes
  - ROMless version available
  - RAM 512/768/1K/1.5K/2K bytes
- 224 general purpose registers available as RAM, accumulators or index pointers (register file)
- 84-pin Plastic Leaded Chip Carrier (64K version only) and 80-pin Plastic Quad Flat Package
- 72/67 fully programmable I/O bits
- 8 external and 1 Non-Maskable Interrupts
- DMA Controller and Programmable Interrupt Handler
- Single Master Serial Peripheral Interface
- Two 16-bit Timers with 8-bit Prescaler, one usable as a Watchdog Timer (software and hardware)
- Three (ST90158) or two (ST90135) 16-bit Multifunction Timers, each with an 8 bit prescaler, 12 operating modes and DMA capabilities
- 8 channel 8-bit Analog to Digital Converter, with Automatic voltage monitoring capabilities and external reference inputs
- Two (ST90158) or one (ST90135) Serial Communication Interfaces with asynchronous, synchronous and DMA capabilities
- Rich Instruction Set with 14 Addressing modes
- Division-by-Zero trap generation



PLCC84 (ST90158 only)



PQFP80 (all)

- Versatile Development Tools, including Assembler, Linker, C-compiler, Archiver, Source Level Debugger and Hardware Emulators with Real-Time Operating System available from Third Parties

DEVICE SUMMARY

DEVICE	Program Memory (Bytes)	RAM (Bytes)	MFT	SCI	PACKAGE
ST90135	16K ROM	512	2	1	PQFP80
	24K ROM	768	2	1	
	32K ROM	1K	2	1	
ST90158	48K ROM	1.5K	3	2	
	64k ROM	2K	3	2	PLCC84/ PQFP80
ST90E158	64K EPROM	2K	3	2	CLCC84/ CQFP80
ST90T158	64K OTP	2K	3	2	PLCC84/ PQFP80
ST90R158	ROMless	2K	3	2	PQFP80

Rev. 2.4

Table of Contents

Table of Contents

Table of Contents

Table of Contents

1 GENERAL INFORMATION

1.1 INTRODUCTION

The ST90158 and ST90135 microcontrollers are developed and manufactured by STMicroelectronics using a proprietary n-well HCMOS process. Their performance derives from the use of a flexible 256-register programming model for ultra-fast context switching and real-time event response. The intelligent on-chip peripherals offload the ST9 core from I/O and data management processing tasks allowing critical application tasks to get the maximum use of core resources. The new-generation ST9 MCU devices now also support low power consumption and low voltage operation for power-efficient and low-cost embedded systems.

1.1.1 ST9+ Core

The advanced Core consists of the Central Processing Unit (CPU), the Register File, the Interrupt and DMA controller, and the Memory Management Unit. The MMU allows addressing of up to 4 Megabytes of program and data mapped into a single linear space.

Four independent buses are controlled by the Core: a 16-bit memory bus, an 8-bit register data bus, an 8-bit register address bus and a 6-bit interrupt/DMA bus which connects the interrupt and DMA controllers in the on-chip peripherals with the core.

This multiple bus architecture makes the ST9 family devices highly efficient for accessing on and off-chip memory and fast exchange of data with the on-chip peripherals.

The general-purpose registers can be used as accumulators, index registers, or address pointers. Adjacent register pairs make up 16-bit registers for addressing or 16-bit processing. Although the ST9 has an 8-bit ALU, the chip handles 16-bit operations, including arithmetic, loads/stores, and memory/register and memory/memory exchanges.

1.1.2 Power Saving Modes

To optimize performance versus power consumption, a range of operating modes can be dynamically selected.

Run Mode. This is the full speed execution mode with CPU and peripherals running at the maximum clock speed delivered by the Phase Locked Loop (PLL) of the Clock Control Unit (CCU).

Slow Mode. Power consumption can be significantly reduced by running the CPU and the peripherals at reduced clock speed using the CPU Prescaler and CCU Clock Divider (PLL not used) or by using the CK_AF external clock.

Wait For Interrupt Mode. The Wait For Interrupt (WFI) instruction suspends program execution until an interrupt request is acknowledged. During WFI, the CPU clock is halted while the peripheral and interrupt controller keep running at a frequency programmable via the CCU. In this mode, the power consumption of the device can be reduced by more than 95% (LP WFI).

Halt Mode. When executing the HALT instruction, and if the Watchdog is not enabled, the CPU and its peripherals stop operation and the I/O ports enter high impedance mode. A reset is necessary to exit from Halt mode.

1.1.3 System Clock

A programmable PLL Clock Generator allows standard 3 to 5 MHz crystals to be used to obtain a large range of internal frequencies up to 16 MHz.

1.1.4 I/O Ports

The I/O lines are grouped into up to nine 8-bit I/O Ports and can be configured on a bit basis to provide timing, status signals, an address/data bus for interfacing to external memory, timer inputs and outputs, analog inputs, external interrupts and serial or parallel I/O.

1.1.5 Multifunction Timers (MFT)

Each multifunction timer has a 16-bit Up/Down counter supported by two 16-bit Compare registers and two 16-bit input capture registers. Timing resolution can be programmed using an 8-bit prescaler. Multibyte transfers between the peripheral and memory are supported by two DMA channels.

1.1.6 Standard Timer (STIM)

The Standard Timer includes a programmable 16-bit down counter and an associated 8-bit prescaler with Single and Continuous counting modes.

1.1.7 Watchdog Timer (WDT)

The Watchdog timer can be used to monitor system integrity. When enabled, it generates a reset after a timeout period unless the counter is refreshed by the application software. For additional security, watchdog function can be enabled by hardware using a specific pin.

1.1.8 Serial Peripheral Interface (SPI)

The SPI bus is used to communicate with external devices via the SPI, or I C bus communication standards. The SPI uses one or two lines for serial data and a synchronous clock signal.

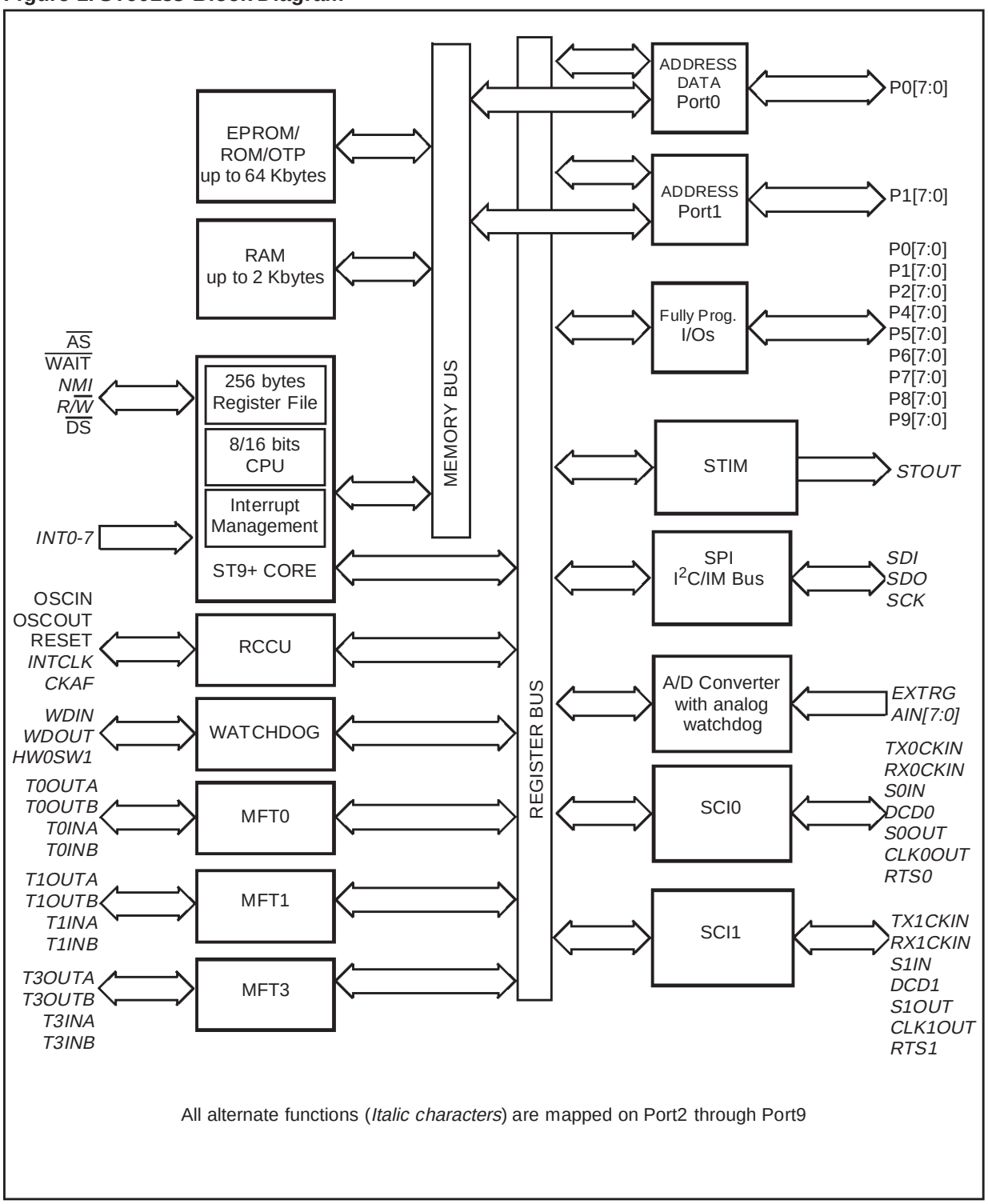
1.1.9 Serial Communications Controllers (SCI)

Each SCI provides a synchronous or asynchronous serial I/O port using two DMA channels. Baud rates and data formats are programmable.

1.1.10 Analog/Digital Converter (ADC)

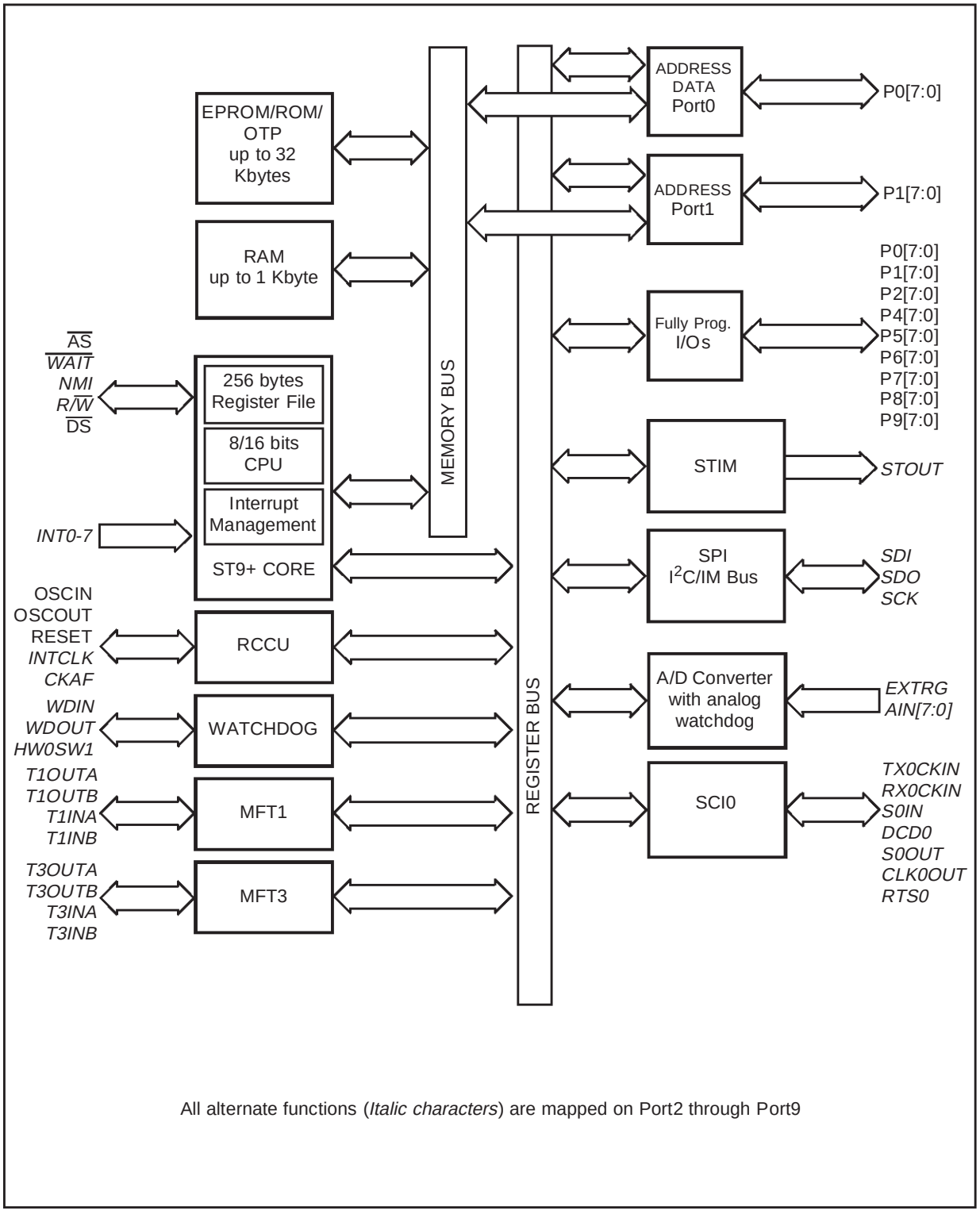
The ADCs provide up to 8 analog inputs with on-chip sample and hold. The analog watchdog generates an interrupt when the input voltage moves out of a preset threshold.

Figure 1. ST90158 Block Diagram



All alternate functions (*Italic characters*) are mapped on Port2 through Port9

Figure 2. ST90135 Block Diagram



All alternate functions (*Italic characters*) are mapped on Port2 through Port9

1.2 PIN DESCRIPTION

$\overline{AS}$: Address Strobe (output, active low, 3-state). Address Strobe is pulsed low once at the beginning of each memory cycle. The rising edge of $\overline{AS}$ indicates that address, Read/Write (R/$\overline{W}$), and Data Memory signals are valid for memory transfers. Under program control, $\overline{AS}$ can be placed in a high-impedance state along with Port 0, Port 1 and Data Strobe ($\overline{DS}$).

$\overline{DS}$: Data Strobe (output, active low, 3-state). Data Strobe provides the timing for data movement to or from Port 0 for each memory transfer. During a write cycle, data out is valid at the leading edge of $\overline{DS}$. During a read cycle, Data In must be valid prior to the trailing edge of $\overline{DS}$. When the ST90158 accesses on-chip memory, $\overline{DS}$ is held high during the whole memory cycle. It can be placed in a high impedance state along with Port 0, Port 1 and $\overline{AS}$.

$\overline{RESET}$: Reset (input, active low). The ST9+ is initialised by the Reset signal. With the deactivation of $\overline{RESET}$, program execution begins from the memory location pointed to by the vector contained in memory locations 00h and 01h.

R/$\overline{W}$: Read/Write (output, 3-state). Read/Write determines the direction of data transfer for external memory transactions. R/$\overline{W}$ is low when writing to external memory, and high for all other transactions. It can be placed in high impedance state along with Port 0, Port 1, $\overline{AS}$ and $\overline{DS}$.

OSCIN, OSCOUT: Oscillator (input and output). These pins connect a parallel-resonant crystal (3 to 5 MHz), or an external source to the on-chip clock oscillator and buffer. OSCIN is the input of the oscillator inverter and internal clock generator; OSCOUT is the output of the oscillator inverter.

HW0_SW1: When connected to V_{DD} through a 1K pull-up resistor, the software watchdog option is selected. When connected to V_{SS} through a 1K pull-down resistor, the hardware watchdog option is selected.

VPP: Programming voltage for EPROM/OTP devices. Must be connected to V_{SS} in user mode through a 10 Kohm resistor.

AV_{DD}: Analog V_{DD} of the Analog to Digital Converter.

AV_{SS}: Analog V_{SS} of the Analog to Digital Converter.

V_{DD}: Main Power Supply Voltage (5V ± 10%).

V_{SS}: Digital Circuit Ground.

P0[7:0], P1[7:0]: (*Input/Output, TTL or CMOS compatible*). 8 lines grouped into I/O ports of 8 bits providing the external memory interface to address the external program memory.

P0[7:0], P1[7:0], P2[7:0], P4[7:0], P5[7:0], P6[7:0], P7[7:0], P8[7:0], P9[7:0]: *I/O Port Lines (Input/Output, TTL or CMOS compatible).* 8 lines grouped into I/O ports of 8 bits, bit programmable under program control as general purpose I/O or as alternate functions

1.2.1 I/O Port Styles

Refer to the I/O Ports section for a description of the I/O port configuration registers.

Table 1. I/O Port Styles

Ports	Weak Pull-up	Port Style
P0	Yes	Standard (TTL/CMOS)
P1	Yes	Standard (TTL/CMOS)
P2	No	Standard (TTL/CMOS)
P4	Yes	Schmitt Trigger
P5	Yes	Schmitt Trigger
P6	No	Standard (TTL/CMOS)
P7	Yes	Schmitt Trigger
P8	Yes	Schmitt Trigger
P9	Yes	Schmitt Trigger

1.2.2 I/O Port Alternate Functions

Each pin of the I/O ports of the ST90158/135 may assume software programmable Alternate Functions as shown in Table 3.

To configure the I/O ports, use the information in Table 1 and the Port Bits Configuration Table in the I/O Ports Chapter.

Inputs:

– If port style = TTL/CMOS, either TTL or CMOS input level can be selected by software.

– If port style = Schmitt trigger, selecting CMOS or TTL input by software has no effect, the input will always be Schmitt trigger.

Weak Pull-Up = This column indicates if a weak pull-up is present or not for bidirectional configuration of the I/O port.

– If WPU = yes, then the WPU can be enabled/disable by software

– If WPU = no, then enabling the WPU by software has no effect

Alternate Functions (AF)= An AF can be selected as follows (more than one AF cannot be assigned to an I/O pin at the same time):

AF Inputs:

– AF is selected implicitly by enabling the corresponding peripheral. Exceptions to this are the ADC inputs which must be explicitly selected as AF by software.

AF Outputs:

– AF output must be selected explicitly by software.

Example 1: SCI data input

AF: S0IN, Port: P9.5, Port Style: Input Schmitt Trigger.

Write the port configuration bits:

P9C2.5=1

P9C1.5=0

P9C0.5=1

Enable the SCI peripheral by software as described in the SCI chapter.

Example 2: SCI data output

AF: S0OUT, Port: P9.4 Output push-pull (configured by software).

Write the port configuration bits:

P9C2.4=0

P9C1.4=1

P9C0.4=1

Example 3: ADC data input

AF: AIN0, Port : P7.0, Port style: does not apply to ADC

Write the port configuration bits:

P7C2.0=1

P7C1.0=1

P7C0.0=1

Example 4: External Memory I/O

AF: AD0, Port : P0.0, Port style: standard (output push-pull).

Write the port configuration bits:

P0C2.0=0

P0C1.0=1

P0C0.0=1

Figure 3. 84-Pin PLCC Pin-Out

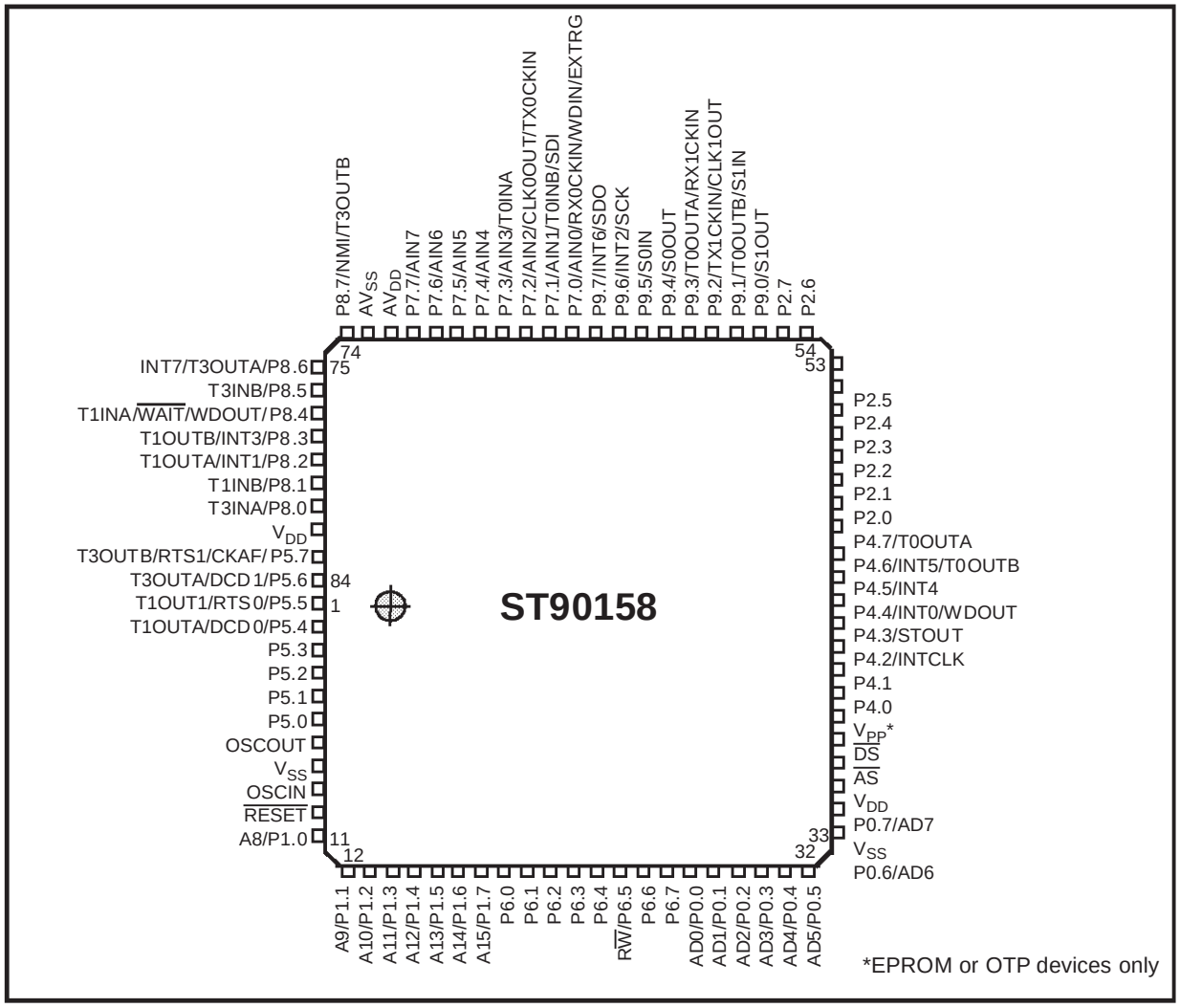


Figure 4. 80-Pin PQFP Pin-Out

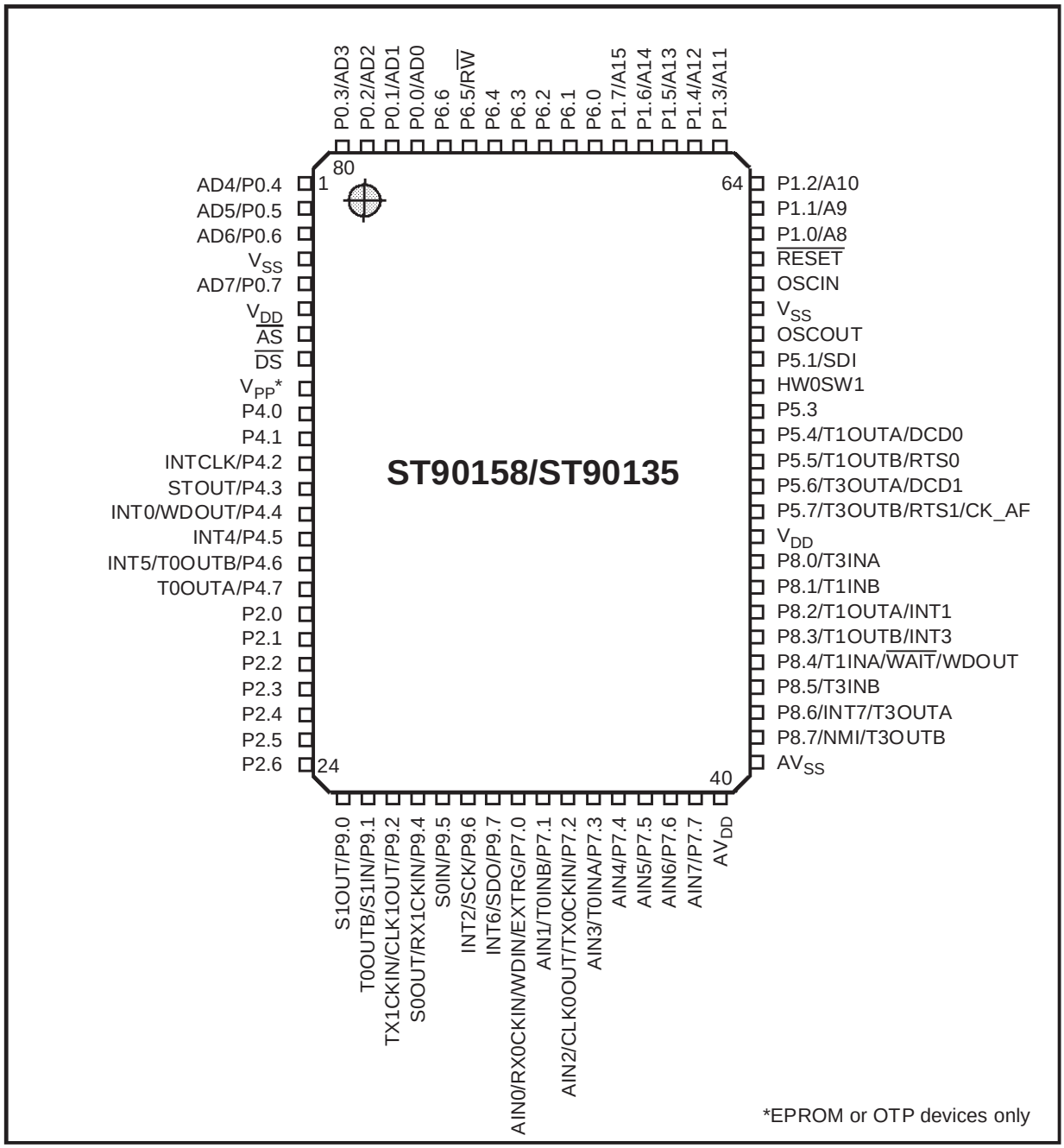


Table 2. PLCC/PQFP Pinout Differences

	PLCC	PQFP
P2.7	Pin 55	not present
P5.0	Pin 6	not present
P5.2	Pin 4	not present
P6.7	Pin 26	not present
P9.3	Pin 59	not present
HW0_SW1 (Watchdog)	Not present (software watchdog only)	Pin 56 (selectable hardware or software watchdog)
RX1CKIN	Port 9.3 (pin 59)	Port 9.4 (pin 28)
SDI	Port 7.1 (pin 65)	Port 5.1 (pin 57)

Table 3. ST90158/ST90135 I/O Port Alternate Function Summary

I/O PORT Port.bit	Name	Function IN/OUT	Alternate Function	Pin Number	
				PLCC	PQFP
P0.0	AD0	I/O	Address/Data bit 0 mux	27	77
P0.1	AD1	I/O	Address/Data bit 1 mux	28	78
P0.2	AD2	I/O	Address/Data bit 2 mux	29	79
P0.3	AD3	I/O	Address/Data bit 3 mux	30	80
P0.4	AD4	I/O	Address/Data bit 4 mux	31	1
P0.5	AD5	I/O	Address/Data bit 5 mux	32	2
P0.6	AD6	I/O	Address/Data bit 6 mux	33	3
P0.7	AD7	I/O	Address/Data bit 7 mux	35	5
P1.0	A8	I/O	Address bit 8	11	62
P1.1	A9	I/O	Address bit 9	12	63
P1.2	A10	I/O	Address bit 10	13	64
P1.3	A11	I/O	Address bit 11	14	65
P1.4	A12	I/O	Address bit 12	15	66
P1.5	A13	I/O	Address bit 13	16	67
P1.6	A14	I/O	Address bit 14	17	68
P1.7	A15	I/O	Address bit 15	18	69
P4.2	INTCLK	0	Internal main Clock	42	12
P4.3	STOUT	O	Standard Timer Output	43	13
P4.4	INT0	I	External Interrupt 0	44	14
P4.4	WDOUT	O	Watchdog Timer output	44	14
P4.5	INT4	I	External interrupt 4	45	15
P4.6	INT5	I	External Interrupt 5	46	16
P4.6	T0OUTB	O	MF Timer 0 Output B [1]	46	16
P4.7	T0OUTA	O	MF Timer 0 Output A [1]	47	17
P5.1	SDI	I	SPI Serial Data In	-	57
P5.4	T1OUTA	O	MF Timer 1 output A	2	54
P5.4	DCD0	I	SCI0 Data Carrier Detect	2	54
P5.5	RTS0	O	SCI0 Request to Send	1	53
P5.5	T1OUTB	O	MF Timer 1 output B	1	53
P5.6	T3OUTA	O	MF Timer 3 output A	84	52
P5.6	DCD1	I	SCI1 Data Carrier Detect [1]	84	52
P5.7	RTS1	O	SCI1 Request to Send [1]	83	51
P5.7	T3OUTB	O	MF Timer 3 output B	83	51
P5.7	CK_AF	I	External Clock Input	83	51
P6.5	R/$\overline{W}$	O	Read/Write	24	75
P7.0	AIN0	I	A/D Analog input 0	64	32
P7.0	RX0CKIN	I	SCI0 Receive Clock input	64	32
P7.0	WDIN	I	T/WD input	64	32
P7.0	EXTRG	I	A/D External Trigger	64	32
P7.1	AIN1	I	A/D Analog input 1	65	33
P7.1	T0INB	I	MF Timer 0 input B [1]	65	33

I/O PORT Port.bit	Name	Function IN/OUT	Alternate Function	Pin Number	
				PLCC	PQFP
P7.1	SDI	I	SPI Serial Data In	65	-
P7.2	AIN2	I	A/D Analog input 2	66	34
P7.2	CLK0OUT	O	SCI0 Byte Sync Clock output	66	34
P7.2	TX0CKIN	I	SCI0 Transmit Clock input	66	34
P7.3	AIN3	I	A/D Analog input 3	67	35
P7.3	T0INA	I	MF Timer 0 input A [1)]	67	35
P7.4	AIN4	I	A/D Analog input 4	68	36
P7.5	AIN5	I	A/D Analog input 5	69	37
P7.6	AIN6	I	A/D Analog input 6	70	38
P7.7	AIN7	I	A/D Analog input 7	71	39
P8.0	T3INA	I	MF Timer 3 input A	81	49
P8.1	T1INB	I	MF Timer 1 input B	80	48
P8.2	INT1	I	External interrupt 1	79	47
P8.2	T1OUTA	O	MF Timer 1 output A	79	47
P8.3	INT3	I	External interrupt 3	78	46
P8.3	T1OUTB	O	MF Timer 1 output B	78	46
P8.4	T1INA	I	MF Timer 1 input A	77	45
P8.4	$\overline{\text{WAIT}}$	I	External Wait input	77	45
P8.4	WDOUT	O	Watchdog Timer output	77	45
P8.5	T3INB	I	MF Timer 3 input B	76	44
P8.6	INT7	I	External interrupt 7	75	43
P8.6	T3OUTA	O	MF Timer 3 output A	75	43
P8.7	NMI	I	Non-Maskable Interrupt	74	42
P8.7	T3OUTB	O	MF Timer 3 output B	74	42
P9.0	S1OUT	O	SCI1 Serial Output [1)]	56	25
P9.1	T0OUTB	O	MF Timer 0 output B [1)]	57	26
P9.1	S1IN	I	SCI1 Serial Input [1)]	57	26
P9.2	CLK1OUT	O	SCI1 Byte Sync Clock output [1)]	58	27
P9.2	TX1CKIN	I	SCI1 Transmit Clock input [1)]	58	27
P9.3	RX1CKIN	I	SCI1 Receive Clock input [1)]	59	-
P9.3	T0OUTA	O	MF Timer 0 output A [1)]	59	-
P9.4	S0OUT	O	SCI0 Serial Output	60	28
P9.4	RX1CKIN	O	SCI1 Receive Clock input [1)]	-	28
P9.5	S0IN	I	SCI0 Serial Input	61	29
P9.6	INT2	I	External interrupt 2	62	30
P9.6	SCK	O	SPI Serial Clock	62	30
P9.7	INT6	I	External interrupt 6	63	31
P9.7	SDO	O	SPI Serial Data Out	63	31

Note 1) Not present on ST90135

2 DEVICE ARCHITECTURE

2.1 CORE ARCHITECTURE

The ST9+ Core or Central Processing Unit (CPU) features a highly optimised instruction set, capable of handling bit, byte (8-bit) and word (16-bit) data, as well as BCD and Boolean formats; 14 addressing modes are available.

Four independent buses are controlled by the Core: a 16-bit Memory bus, an 8-bit Register data bus, an 8-bit Register address bus and a 6-bit Interrupt/DMA bus which connects the interrupt and DMA controllers in the on-chip peripherals with the Core.

This multiple bus architecture affords a high degree of pipelining and parallel operation, thus making the ST9+ family devices highly efficient, both for numerical calculation, data handling and with regard to communication with on-chip peripheral resources.

2.2 MEMORY SPACES

There are two separate memory spaces:

– The Register File, which comprises 240 8-bit registers, arranged as 15 groups (Group 0 to E), each containing sixteen 8-bit registers plus up to 64 pages of 16 registers mapped in Group F, which hold data and control bits for the on-chip peripherals and I/Os.

– A single linear memory space accommodating both program and data. All of the physically separate memory areas, including the internal ROM, internal RAM and external memory are mapped in this common address space. A total addressable memory space of 4 Mbytes is available. This address space is arranged as 64 segments of 64 Kbytes. Each segment is further subdivided into four pages of 16 Kbytes, as illustrated in Figure 5. A Memory Management Unit uses a set of pointer registers to address a 22-bit memory field using 16-bit address-based instructions.

2.2.1 Register File

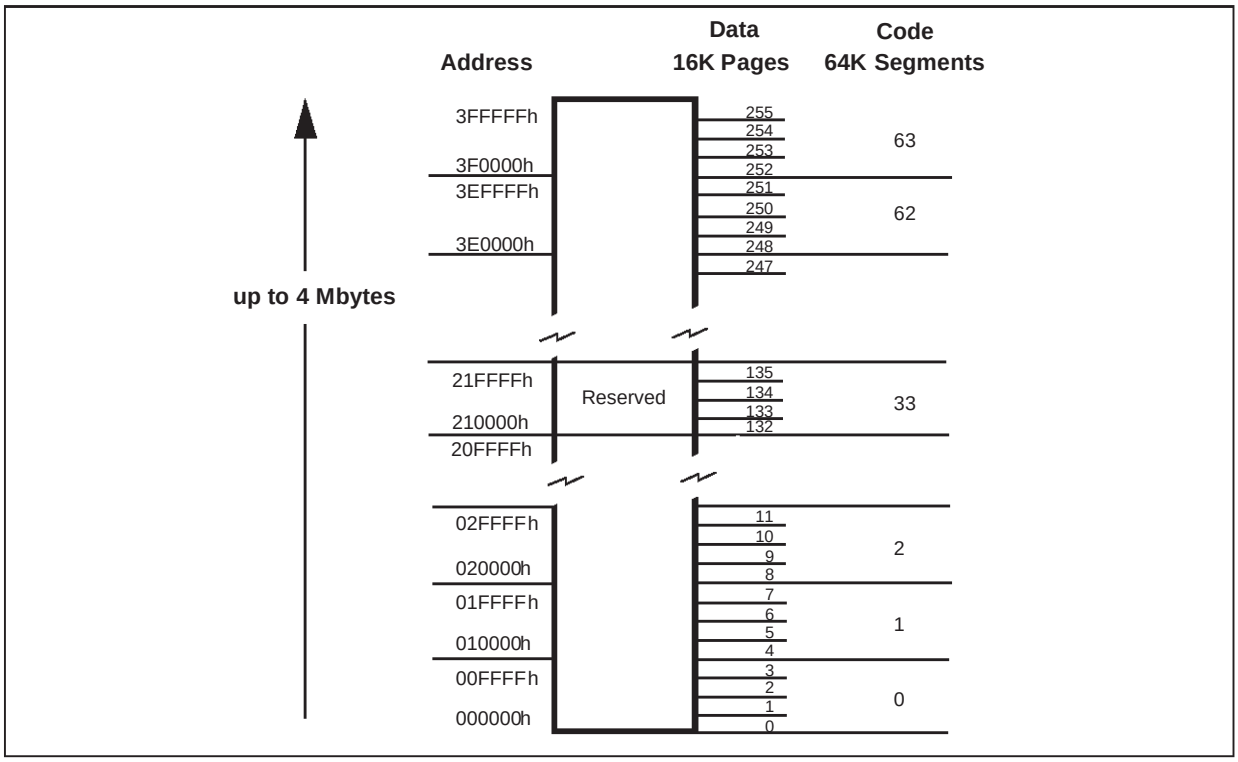
The Register File consists of (see Figure 6):

– 224 general purpose registers (Group 0 to D, registers R0 to R223)

– 16 system registers in the System Group (Group E, registers R224 to R239)

– Up to 64 pages, depending on device configuration, each containing up to 16 registers, mapped to Group F (registers R240 to R255).

Figure 5. Single Program and Data Memory Address Space



MEMORY SPACES (Cont'd)

Figure 6. Register Groups

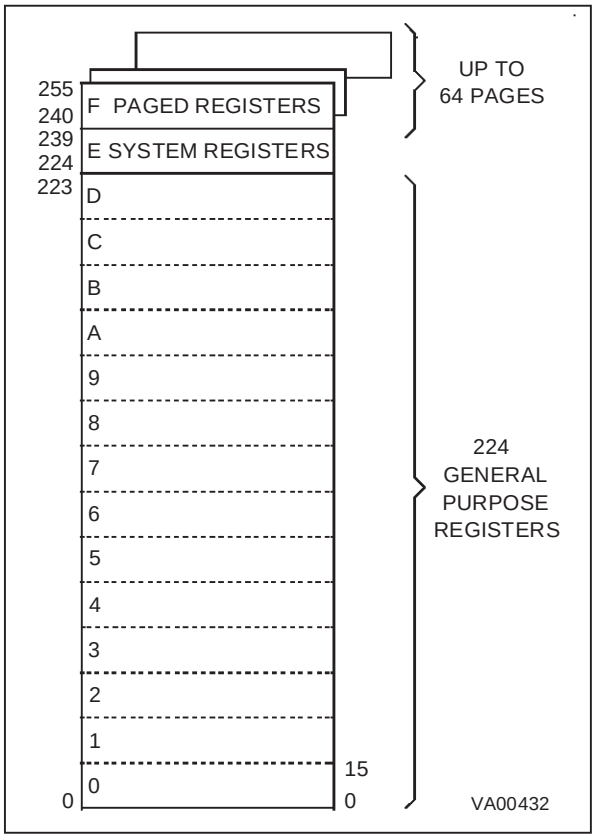


Figure 7. Page Pointer for Group F mapping

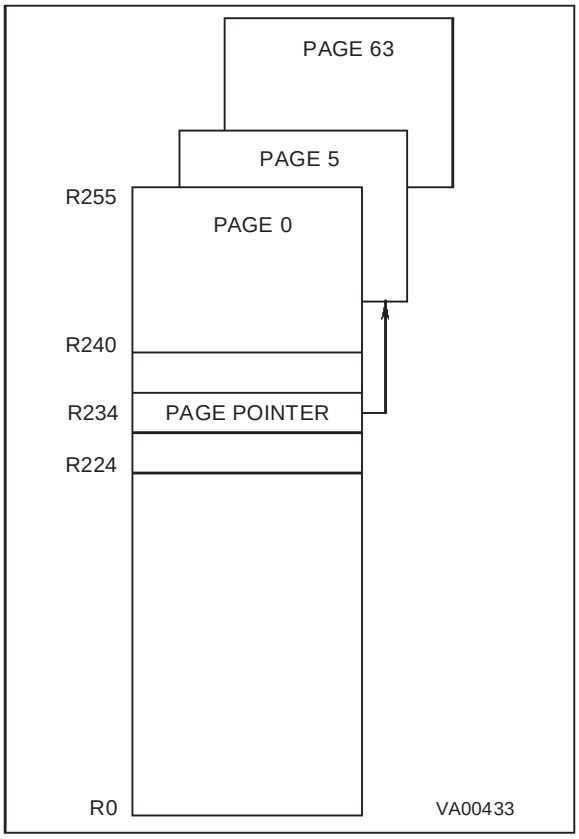
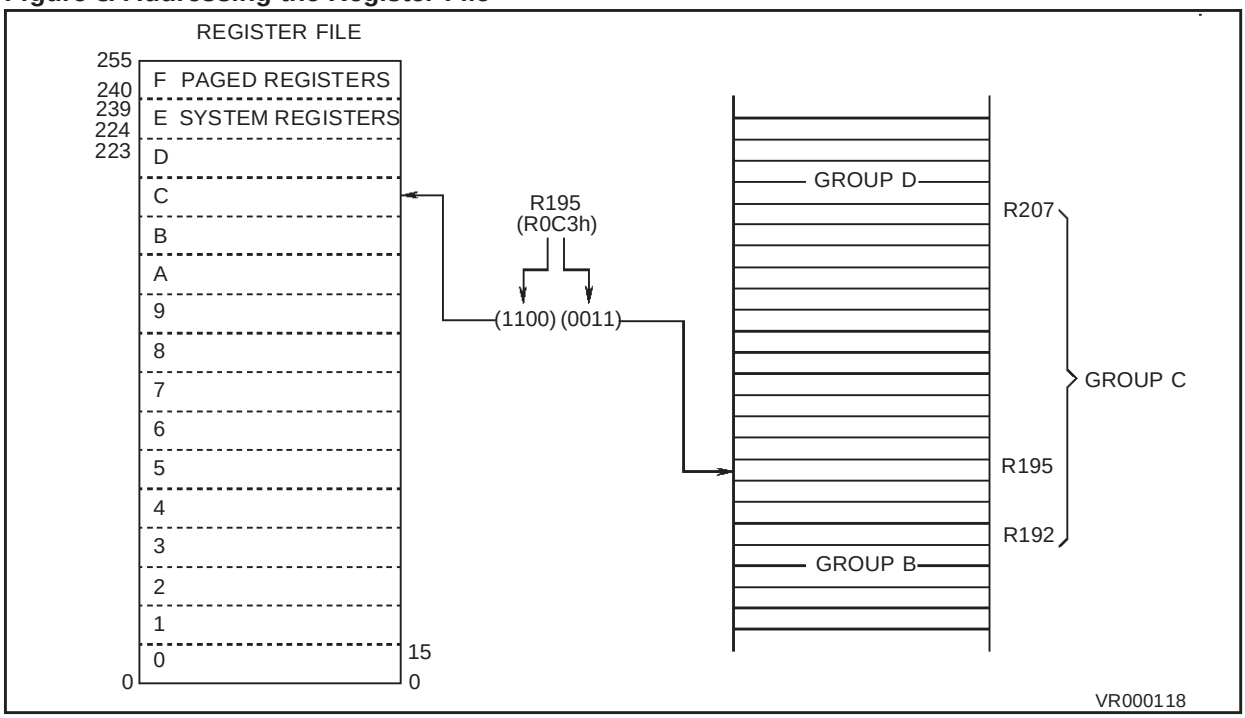


Figure 8. Addressing the Register File



MEMORY SPACES (Cont'd)

2.2.2 Register Addressing

Register File registers, including Group F paged registers (but excluding Group D), may be addressed explicitly by means of a decimal, hexadecimal or binary address; thus `R231`, `RE7h` and `R11100111b` represent the same register. Group D registers can only be addressed in Working Register mode.

Note that an upper case "R" is used to denote this direct addressing mode.

Working Registers

Certain types of instruction require that registers be specified in the form "rx", where x is in the range 0 to 15: these are known as Working Registers.

Note that a lower case "r" is used to denote this indirect addressing mode.

Two addressing schemes are available: a single group of 16 working registers, or two separately mapped groups, each consisting of 8 working registers. These groups may be mapped starting at any 8 or 16 byte boundary in the register file by means of dedicated pointer registers. This technique is described in more detail in Section 2.3.3 Register Pointing Techniques, and illustrated in Figure 9 and in Figure 10.

System Registers

The 16 registers in Group E (R224 to R239) are System registers and may be addressed using any of the register addressing modes. These registers are described in greater detail in Section 2.3 SYSTEM REGISTERS.

Paged Registers

Up to 64 pages, each containing 16 registers, may be mapped to Group F. These are addressed using any register addressing mode, in conjunction with the Page Pointer register, R234, which is one of the System registers. This register selects the page to be mapped to Group F and, once set, does not need to be changed if two or more registers on the same page are to be addressed in succession.

Therefore if the Page Pointer, R234, is set to 5, the instructions:

```
spp #5
ld R242, r4
```

will load the contents of working register r4 into the third register of page 5 (R242).

These paged registers hold data and control information relating to the on-chip peripherals, each peripheral always being associated with the same pages and registers to ensure code compatibility between ST9+ devices. The number of these registers therefore depends on the peripherals which are present in the specific ST9+ family device. In other words, pages only exist if the relevant peripheral is present.

Table 4. Register File Organization

Hex. Address	Decimal Address	Function	Register File Group
F0-FF	240-255	Paged Registers	Group F
E0-EF	224-239	System Registers	Group E
D0-DF	208-223	General Purpose Registers	Group D
C0-CF	192-207		Group C
B0-BF	176-191		Group B
A0-AF	160-175		Group A
90-9F	144-159		Group 9
80-8F	128-143		Group 8
70-7F	112-127		Group 7
60-6F	96-111		Group 6
50-5F	80-95		Group 5
40-4F	64-79		Group 4
30-3F	48-63		Group 3
20-2F	32-47		Group 2
10-1F	16-31		Group 1
00-0F	00-15		Group 0

2.3 SYSTEM REGISTERS

The System registers are listed inTable 5 System Registers (Group E). They are used to perform all the important system settings. Their purpose is described in the following pages. Refer to the chapter dealing with I/O for a description of the PORT[5:0] Data registers.

Table 5. System Registers (Group E)

R239 (EFh)	SSPLR
R238 (EEh)	SSPHR
R237 (EDh)	USPLR
R236 (ECh)	USPHR
R235 (EBh)	MODE REGISTER
R234 (EAh)	PAGE POINTER REGISTER
R233 (E9h)	REGISTER POINTER 1
R232 (E8h)	REGISTER POINTER 0
R231 (E7h)	FLAG REGISTER
R230 (E6h)	CENTRAL INT. CNTL REG
R229 (E5h)	PORT5 DATA REG.
R228 (E4h)	PORT4 DATA REG.
R227 (E3h)	PORT3 DATA REG.
R226 (E2h)	PORT2 DATA REG.
R225 (E1h)	PORT1 DATA REG.
R224 (E0h)	PORT0 DATA REG.

2.3.1 Central Interrupt Control Register

Please refer to the "INTERRUPT" and "DMA" chapters for a detailed description of the ST9 interrupt philosophy.

CENTRAL INTERRUPT CONTROL REGISTER (CICR)
R230 - Read/Write
Register Group: E (System)
Reset Value: 1000 0111 (87h)

7							0
GCEN	TLIP	TLI	IEN	IAM	CPL2	CPL1	CPL0

Bit 7 = **GCEN**: *Global Counter Enable.*
This bit is the Global Counter Enable of the Multifunction Timers. The GCEN bit is ANDed with the CE bit in the TCR Register (only in devices featuring the MFT Multifunction Timer) in order to enable the Timers when both bits are set. This bit is set after the Reset cycle.

Note: If an MFT is not included in the ST9 device, then this bit has no effect.

Bit 6 = **TLIP**: *Top Level Interrupt Pending.*
This bit is set by hardware when a Top Level Interrupt Request is recognized. This bit can also be set by software to simulate a Top Level Interrupt Request.
0: No Top Level Interrupt pending
1: Top Level Interrupt pending

Bit 5 = **TLI**: *Top Level Interrupt bit*
0: Top Level Interrupt is acknowledged depending on the TLNM bit in the NICR Register.
1: Top Level Interrupt is acknowledged depending on the IEN and TLNM bits in the NICR Register (described in the Interrupt chapter).

Bit 4 = **IEN**: *Enable Interrupt.*
This bit is cleared by interrupt acknowledgement, and set by interrupt return (`iret`). IEN is modified implicitly by `iret`, `ei` and `di` instructions or by an interrupt acknowledge cycle. It can also be explicitly written by the user, but only when no interrupt is pending. Therefore, the user should execute a `di` instruction (or guarantee by other means that no interrupt request can arrive) before any write operation to the CICR register.
0: Disable all interrupts except Top Level Interrupt.
1: Enable Interrupts

Bit 3 = **IAM**: *Interrupt Arbitration Mode.*
This bit is set and cleared by software to select the arbitration mode.
0: Concurrent Mode
1: Nested Mode.

Bit 2:0 = **CPL[2:0]**: *Current Priority Level.*
These three bits record the priority level of the routine currently running (i.e. the Current Priority Level, CPL). The highest priority level is represented by 000, and the lowest by 111. The CPL bits can be set by hardware or software and provide the reference according to which subsequent interrupts are either left pending or are allowed to interrupt the current interrupt service routine. When the current interrupt is replaced by one of a higher priority, the current priority value is automatically stored until required in the NICR register.

SYSTEM REGISTERS (Cont'd)

2.3.2 Flag Register

The Flag Register contains 8 flags which indicate the CPU status. During an interrupt, the flag register is automatically stored in the system stack area and recalled at the end of the interrupt service routine, thus returning the CPU to its original status.

This occurs for all interrupts and, when operating in nested mode, up to seven versions of the flag register may be stored.

FLAG REGISTER (FLAGR)
R231- Read/Write
Register Group: E (System)
Reset value: 0000 0000 (00h)

7							0
C	Z	S	V	DA	H	-	DP

Bit 7 = **C**: *Carry Flag*.
The carry flag is affected by:
Addition (`add, addw, adc, adcw`),
Subtraction (`sub, subw, sbc, sbcw`),
Compare (`cp, cpw`),
Shift Right Arithmetic (`sra, sraw`),
Shift Left Arithmetic (`sla, slaw`),
Swap Nibbles (`swap`),
Rotate (`rrc, rrcw, rlc, rlcw, ror, rol`),
Decimal Adjust (`da`),
Multiply and Divide (`mul, div, divws`).

When set, it generally indicates a carry out of the most significant bit position of the register being used as an accumulator (bit 7 for byte operations and bit 15 for word operations).

The carry flag can be set by the Set Carry Flag (`scf`) instruction, cleared by the Reset Carry Flag (`rcf`) instruction, and complemented by the Complement Carry Flag (`ccf`) instruction.

Bit 6 = **Z:** *Zero Flag*. The Zero flag is affected by:
Addition (`add, addw, adc, adcw`),
Subtraction (`sub, subw, sbc, sbcw`),
Compare (`cp, cpw`),
Shift Right Arithmetic (`sra, sraw`),
Shift Left Arithmetic (`sla, slaw`),
Swap Nibbles (`swap`),
Rotate (`rrc, rrcw, rlc, rlcw, ror, rol`),
Decimal Adjust (`da`),
Multiply and Divide (`mul, div, divws`),
Logical (`and, andw, or, orw, xor, xorw, cpl`),
Increment and Decrement (`inc, incw, dec, decw`),
Test (`tm, tmw, tcm, tcmw, btset`).

In most cases, the Zero flag is set when the contents of the register being used as an accumulator become zero, following one of the above operations.

Bit 5 = **S**: *Sign Flag*.
The Sign flag is affected by the same instructions as the Zero flag.

The Sign flag is set when bit 7 (for a byte operation) or bit 15 (for a word operation) of the register used as an accumulator is one.

Bit 4 = **V**: *Overflow Flag*.
The Overflow flag is affected by the same instructions as the Zero and Sign flags.

When set, the Overflow flag indicates that a two's-complement number, in a result register, is in error, since it has exceeded the largest (or is less than the smallest), number that can be represented in two's-complement notation.

Bit 3 = **DA**: *Decimal Adjust Flag*.
The DA flag is used for BCD arithmetic. Since the algorithm for correcting BCD operations is different for addition and subtraction, this flag is used to specify which type of instruction was executed last, so that the subsequent Decimal Adjust (`da`) operation can perform its function correctly. The DA flag cannot normally be used as a test condition by the programmer.

Bit 2 = **H**: *Half Carry Flag.*
The H flag indicates a carry out of (or a borrow into) bit 3, as the result of adding or subtracting two 8-bit bytes, each representing two BCD digits. The H flag is used by the Decimal Adjust (`da`) instruction to convert the binary result of a previous addition or subtraction into the correct BCD result. Like the DA flag, this flag is not normally accessed by the user.

Bit 1 = Reserved bit (must be 0).

Bit 0 = **DP**: *Data/Program Memory Flag*.
This bit indicates the memory area addressed. Its value is affected by the Set Data Memory (`sdm`) and Set Program Memory (`spm`) instructions. Refer to the Memory Management Unit for further details.

If the bit is set, data is accessed using the Data Pointers (DPRs registers), otherwise it is pointed to by the Code Pointer (CSR register); therefore, the user initialization routine must include a `Sdm` instruction. Note that code is always pointed to by the Code Pointer (CSR).

Note: In the ST9+, the DP flag is only for compatibility with software developed for the first generation of ST9 devices. With the single memory addressing space, its use is now redundant. It must be kept to 1 with a `Sdm` instruction at the beginning of the program to ensure a normal use of the different memory pointers.

2.3.3 Register Pointing Techniques

Two registers within the System register group, are used as pointers to the working registers. Register Pointer 0 (R232) may be used on its own as a single pointer to a 16-register working space, or in conjunction with Register Pointer 1 (R233), to point to two separate 8-register spaces.

For the purpose of register pointing, the 16 register groups of the register file are subdivided into 32 8-register blocks. The values specified with the Set Register Pointer instructions refer to the blocks to be pointed to in twin 8-register mode, or to the lower 8-register block location in single 16-register mode.

The Set Register Pointer instructions `srp`, `srp0` and `srp1` automatically inform the CPU whether the Register File is to operate in single 16-register mode or in twin 8-register mode. The `srp` instruction selects the single 16-register group mode and specifies the location of the lower 8-register block, while the `srp0` and `srp1` instructions automatically select the twin 8-register group mode and specify the locations of each 8-register block.

There is no limitation on the order or position of these register groups, other than that they must start on an 8-register boundary in twin 8-register mode, or on a 16-register boundary in single 16-register mode.

The block number should always be an even number in single 16-register mode. The 16-register group will always start at the block whose number is the nearest even number equal to or lower than the block number specified in the `srp` instruction. Avoid using odd block numbers, since this can be confusing if twin mode is subsequently selected.

Thus:

`srp #3` will be interpreted as `srp #2` and will allow using R16 ..R31 as r0 .. r15.

In single 16-register mode, the working registers are referred to as `r0` to `r15`. In twin 8-register mode, registers `r0` to `r7` are in the block pointed to by RP0 (by means of the `srp0` instruction), while registers `r8` to `r15` are in the block pointed to by RP1 (by means of the `srp1` instruction).

Caution: *Group D registers can only be accessed as working registers using the Register Pointers, or by means of the Stack Pointers. They cannot be addressed explicitly in the form "Rxx".*

SYSTEM REGISTERS (Cont'd)

POINTER 0 REGISTER (RP0)
R232 - Read/Write
Register Group: E (System)
Reset Value: xxxx xx00 (xxh)

7							0
RG4	RG3	RG2	RG1	RG0	RPS	0	0

Bit 7:3 = **RG[4:0]**: *Register Group number.*
These bits contain the number (in the range 0 to 31) of the register block specified in the `srp0` or `srp` instructions. In single 16-register mode the number indicates the lower of the two 8-register blocks to which the 16 working registers are to be mapped, while in twin 8-register mode it indicates the 8-register block to which `r0` to `r7` are to be mapped.

Bit 2 = **RPS**: *Register Pointer Selector.*
This bit is set by the instructions `srp0` and `srp1` to indicate that the twin register pointing mode is selected. The bit is reset by the `srp` instruction to indicate that the single register pointing mode is selected.
0: Single register pointing mode
1: Twin register pointing mode

Bit 1:0: Reserved. Forced by hardware to zero.

POINTER 1 REGISTER (RP1)
R233 - Read/Write
Register Group: E (System)
Reset Value: xxxx xx00 (xxh)

7							0
RG4	RG3	RG2	RG1	RG0	RPS	0	0

This register is only used in the twin register pointing mode. When using the single register pointing mode, or when using only one of the twin register groups, the RP1 register must be considered as RESERVED and may NOT be used as a general purpose register.

Bit 7:3 = **RG[4:0]:** *Register Group number.* These bits contain the number (in the range 0 to 31) of the 8-register block specified in the `srp1` instruction, to which `r8` to `r15` are to be mapped.

Bit 2 = **RPS**: *Register Pointer Selector.*
This bit is set by the `srp0` and `srp1` instructions to indicate that the twin register pointing mode is selected. The bit is reset by the `srp` instruction to indicate that the single register pointing mode is selected.
0: Single register pointing mode
1: Twin register pointing mode

Bit 1:0: Reserved. Forced by hardware to zero..

SYSTEM REGISTERS (Cont'd)

Figure 9. Pointing to a single group of 16 registers

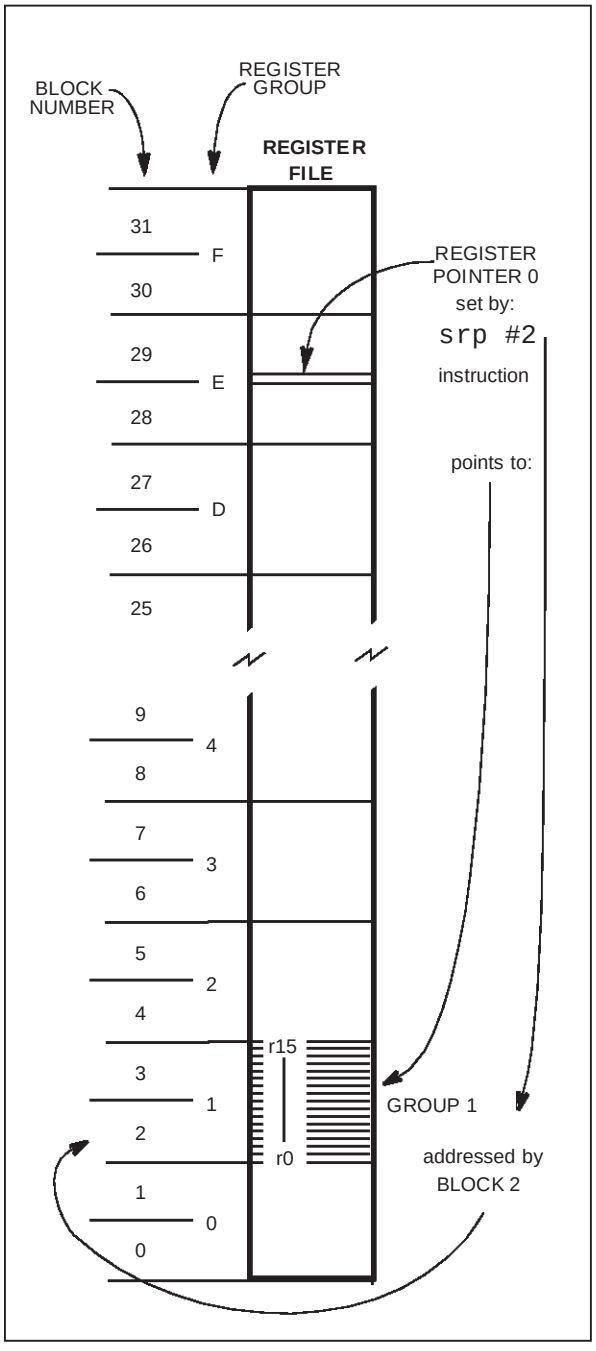
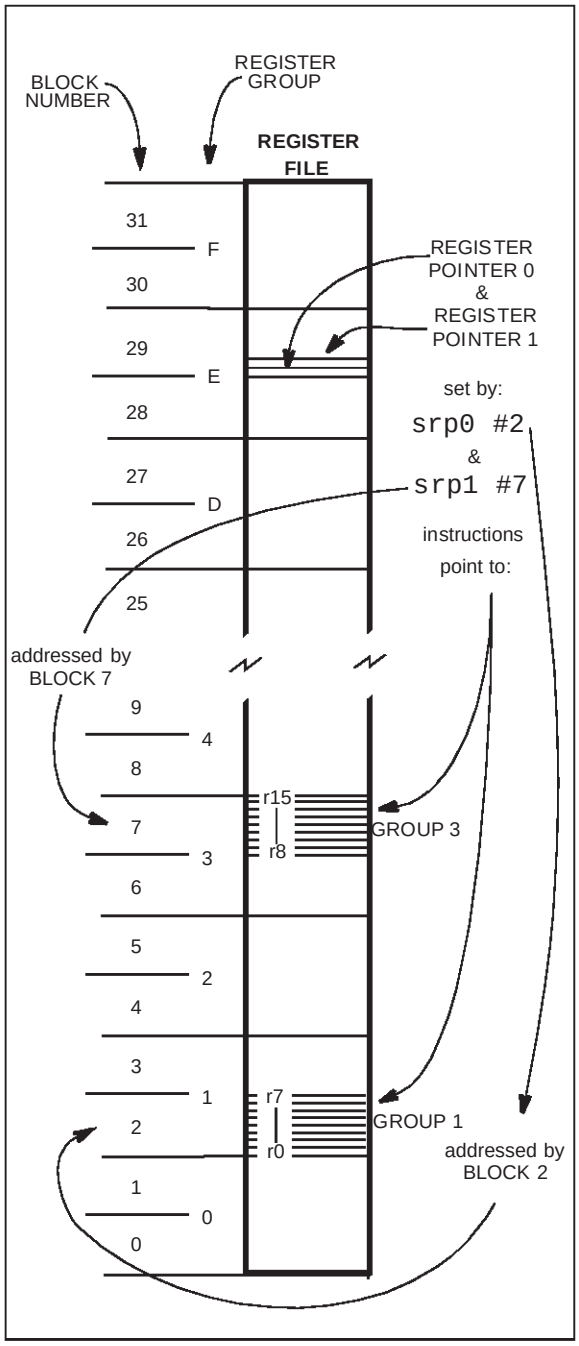


Figure 10. Pointing to two groups of 8 registers



SYSTEM REGISTERS (Cont'd)

2.3.4 Paged Registers

Up to 64 pages, each containing 16 registers, may be mapped to Group F. These paged registers hold data and control information relating to the on-chip peripherals, each peripheral always being associated with the same pages and registers to ensure code compatibility between ST9+ devices. The number of these registers depends on the peripherals present in the specific ST9 device. In other words, pages only exist if the relevant peripheral is present.

The paged registers are addressed using the normal register addressing modes, in conjunction with the Page Pointer register, R234, which is one of the System registers. This register selects the page to be mapped to Group F and, once set, does not need to be changed if two or more registers on the same page are to be addressed in succession.

Thus the instructions:

```
spp #5
ld R242, r4
```

will load the contents of working register r4 into the third register of page 5 (R242).

Warning: *During an interrupt, the PPR register is not saved automatically in the stack. If needed, it should be saved/restored by the user within the interrupt routine.*

PAGE POINTER REGISTER (PPR)
R234 - Read/Write
Register Group: E (System)
Reset value: xxxx xx00 (xxh)

7							0
PP5	PP4	PP3	PP2	PP1	PP0	0	0

Bit 7:2 = **PP[5:0]**: *Page Pointer.*
These bits contain the number (in the range 0 to 63) of the page specified in the spp instruction. Once the page pointer has been set, there is no need to refresh it unless a different page is required.

Bit 1:0: Reserved. Forced by hardware to 0.

2.3.5 Mode Register

The Mode Register allows control of the following operating parameters:

– Selection of internal or external System and User Stack areas,
– Management of the clock frequency,
– Enabling of Bus request and Wait signals when interfacing to external memory.

MODE REGISTER (MODER)
R235 - Read/Write
Register Group: E (System)
Reset value: 1110 0000 (E0h)

7							0
SSP	USP	DIV2	PRS2	PRS1	PRS0	BRQEN	HIMP

Bit 7 = **SSP**: *System Stack Pointer.*
This bit selects an internal or external System Stack area.
0: External system stack area, in memory space.
1: Internal system stack area, in the Register File (reset state).

Bit 6 = **USP**: *User Stack Pointer.*
This bit selects an internal or external User Stack area.
0: External user stack area, in memory space.
1: Internal user stack area, in the Register File (reset state).

Bit 5 = **DIV2**: *OSCIN Clock Divided by 2*
This bit controls the divide-by-2 circuit operating on OSCIN.
0: Clock divided by 1
1: Clock divided by 2

Bit 4:2 = **PRS2-PRS0**: *CPUCLK Prescaler.*
These bits load the prescaler division factor for the internal clock (INTCLK). The prescaler factor selects the internal clock frequency, which can be divided by a factor from 1 to 8. Refer to the Reset and Clock Control chapter for further information.

Bit 1 = **BRQEN**: *Bus Request Enable.*
0: Bus Request disabled
1: Bus Request enabled on the BUSREQ pin (where available).

Bit 0 = **HIMP**: *High Impedance Enable.*
When any of Ports 0, 1, 2 or 6 depending on device configuration, are programmed as Address and Data lines to interface external Memory, these lines and the Memory interface control lines (AS,

SYSTEM REGISTERS (Cont'd)

DS, R/W) can be forced into the High Impedance state by setting the HIMP bit. When this bit is reset, it has no effect.

Setting the HIMP bit is recommended for noise reduction when only internal Memory is used.

If Port 1 and/or 2 are declared as an address AND as an I/O port (for example: P10... P14 = Address, and P15... P17 = I/O), the HIMP bit has no effect on the I/O lines.

2.3.6 Stack Pointers

Two separate, double-register stack pointers are available: the System Stack Pointer and the User Stack Pointer, both of which can address registers or memory.

The stack pointers point to the "bottom" of the stacks which are filled using the push commands and emptied using the pop commands. The stack pointer is automatically pre-decremented when data is "pushed" in and post-incremented when data is "popped" out.

The push and pop commands used to manage the System Stack may be addressed to the User Stack by adding the suffix 'u". To use a stack instruction for a word, the suffix 'w" is added. These suffixes may be combined.

When bytes (or words) are "popped" out from a stack, the contents of the stack locations are unchanged until fresh data is loaded. Thus, when data is "popped" from a stack area, the stack contents remain unchanged.

Note: Instructions such as: `pushuw RR236` or `pushw RR238`, as well as the corresponding `pop` instructions (where R236 & R237, and R238 & R239 are themselves the user and system stack pointers respectively), must not be used, since the pointer values are themselves automatically changed by the `push` or `pop` instruction, thus corrupting their value.

System Stack

The System Stack is used for the temporary storage of system and/or control data, such as the Flag register and the Program counter.

The following automatically push data onto the System Stack:

– **Interrupts**

When entering an interrupt, the PC and the Flag Register are pushed onto the System Stack. If the ENCSR bit in the EMR2 register is set, then the Code Segment Register is also pushed onto the System Stack.

– **Subroutine Calls**

When a `call` instruction is executed, only the PC is pushed onto stack, whereas when `acalls` instruction (call segment) is executed, both the PC and the Code Segment Register are pushed onto the System Stack.

– **Link Instruction**

The `link` or `linku` instructions create a C language stack frame of user-defined length in the System or User Stack.

All of the above conditions are associated with their counterparts, such as return instructions, which pop the stored data items off the stack.

User Stack

The User Stack provides a totally user-controlled stacking area.

The User Stack Pointer consists of two registers, R236 and R237, which are both used for addressing a stack in memory. When stacking in the Register File, the User Stack Pointer High Register, R236, becomes redundant but must be considered as reserved.

Stack Pointers

Both System and User stacks are pointed to by double-byte stack pointers. Stacks may be set up in RAM or in the Register File. Only the lower byte will be required if the stack is in the Register File. The upper byte must then be considered as reserved and must not be used as a general purpose register.

The stack pointer registers are located in the System Group of the Register File, this is illustrated in Table 5 System Registers (Group E)

Stack location

Care is necessary when managing stacks as there is no limit to stack sizes apart from the bottom of any address space in which the stack is placed. Consequently programmers are advised to use a stack pointer value as high as possible, particularly when using the Register File as a stacking area.

Group D is a good location for a stack in the Register File, since it is the highest available area. The stacks may be located anywhere in the first 14 groups of the Register File (internal stacks) or in RAM (external stacks).

Note. Stacks must not be located in the Paged Register Group or in the System Register Group.

SYSTEM REGISTERS (Cont'd)

USER STACK POINTER HIGH REGISTER (USPHR)
R236 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
USP15	USP14	USP13	USP12	USP11	USP10	USP9	USP8

USER STACK POINTER LOW REGISTER (USPLR)
R237 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
USP7	USP6	USP5	USP4	USP3	USP2	USP1	USP0

SYSTEM STACK POINTER HIGH REGISTER (SSPHR)
R238 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
SSP15	SSP14	SSP13	SSP12	SSP11	SSP10	SSP9	SSP8

SYSTEM STACK POINTER LOW REGISTER (SSPLR)
R239 - Read/Write
Register Group: E (System)
Reset value: undefined

7							0
SSP7	SSP6	SSP5	SSP4	SSP3	SSP2	SSP1	SSP0

Figure 11. Internal Stack Mode

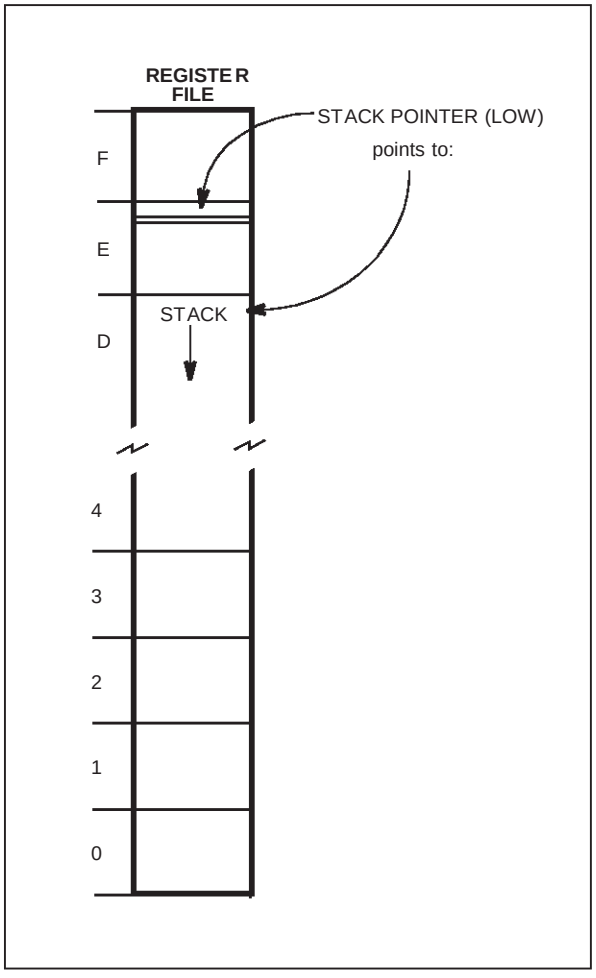
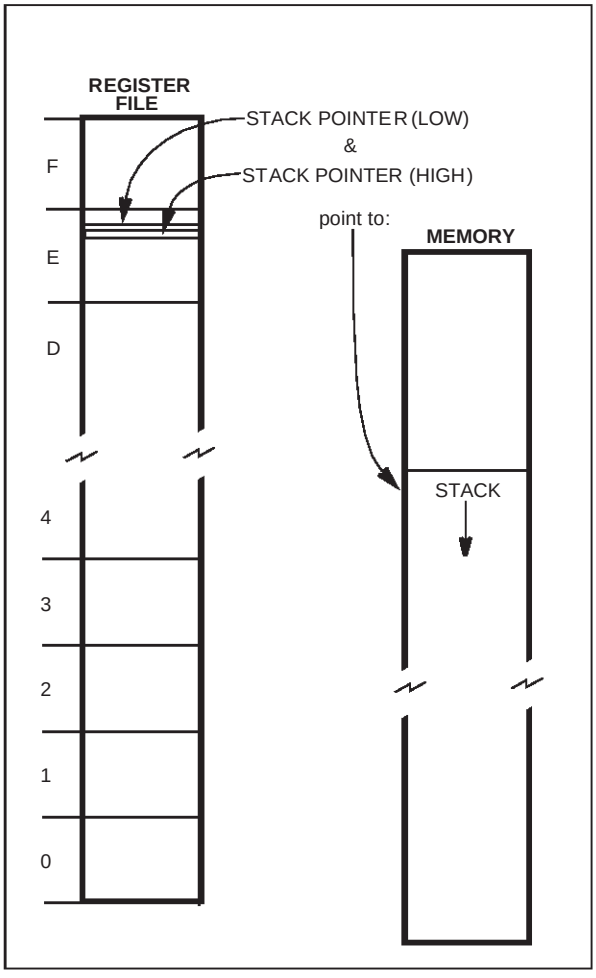


Figure 12. External Stack Mode



2.4 MEMORY ORGANIZATION

Code and data are accessed within the same linear address space. All of the physically separate memory areas, including the internal ROM, internal RAM and external memory are mapped in a common address space.

The ST9+ provides a total addressable memory space of 4 Mbytes. This address space is arranged as 64 segments of 64 Kbytes; each segment is again subdivided into four 16 Kbyte pages.

The mapping of the various memory areas (internal RAM or ROM, external memory) differs from device to device. Each 64-Kbyte physical memory segment is mapped either internally or externally; if the memory is internal and smaller than 64 Kbytes, the remaining locations in the 64-Kbyte segment are not used (reserved).

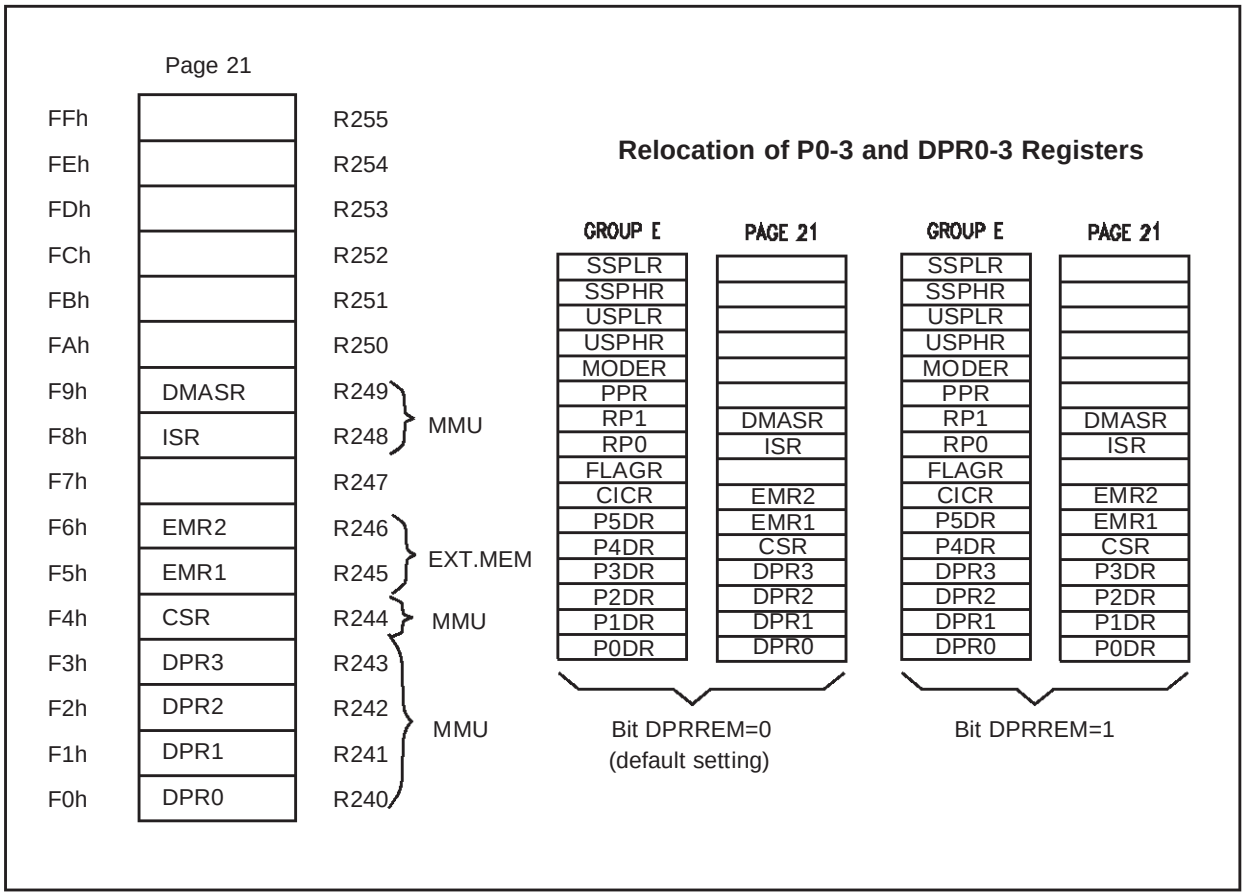
Refer to the Register and Memory Map Chapter for more details on the memory map.

2.5 MEMORY MANAGEMENT UNIT

The CPU Core includes a Memory Management Unit (MMU) which allows the addressing space to be extended to 4 Mbytes.

The MMU is controlled by 7 registers and 2 bits (ENCSR and DPRREM) present in EMR2, which may be written and read by the user program. These registers are mapped within group F, Page 21 of the Register File. The 7 registers may be sub-divided into 2 main groups: a first group of four 8-bit registers (DPR0-3), and a second group of three 6-bit registers (CSR, ISR, and DMASR). The first group is used to extend the address during Data Memory access (DPR0-3). The second is used to manage Program and Data Memory accesses during Code execution (CSR), Interrupts Service Routines (ISR or CSR), and DMA transfers (DMASR or ISR).

Figure 13. Page 21 Registers



2.6 ADDRESS SPACE EXTENSION

To manage 4 Mbytes of addressing space it is necessary to have 22 address bits. The MMU adds 6 bits to the usual 16-bit address, thus translating a 16-bit virtual address into a 22-bit physical address. There are 2 different ways to do this depending on the memory involved and on the operation being performed.

2.6.1 Addressing 16-Kbyte Pages

This extension mode is implicitly used to address Data memory space if no DMA is being performed.

The Data memory space is divided into 4 pages of 16 Kbytes. Each one of the four 8-bit registers (DPR0-3, Data Page Registers) selects a different 16-Kbyte page. The DPR registers allow access to the entire memory space which contains 256 pages of 16 Kbytes.

Data paging is performed by extending the 14 LSB of the 16-bit address with the contents of a DPR register. The two MSBs of the 16-bit address are interpreted as the identification number of the DPR register to be used. Therefore, the DPR registers are involved in the following virtual address ranges:

DPR0: from 0000h to 3FFFh;

DPR1: from 4000h to 7FFFh;

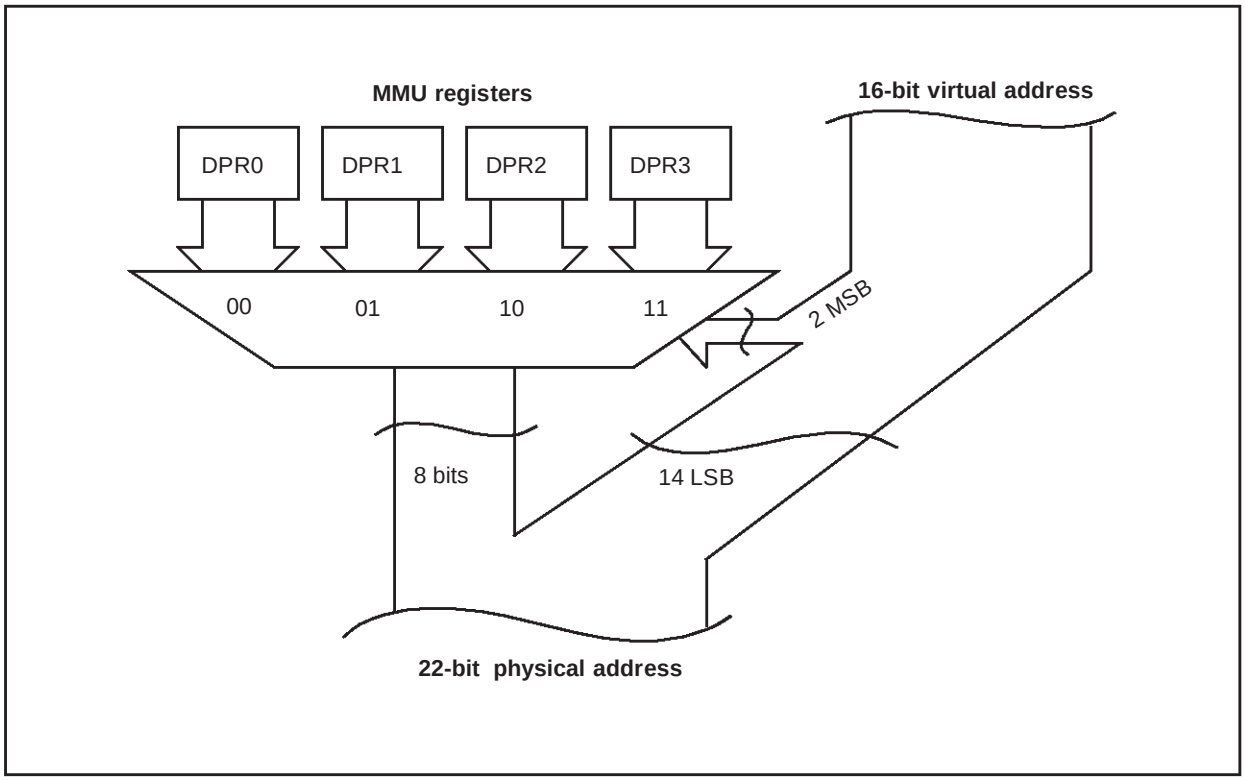
DPR2: from 8000h to BFFFh;

DPR3: from C000h to FFFFh.

The contents of the selected DPR register specify one of the 256 possible data memory pages. This 8-bit data page number, in addition to the remaining 14-bit page offset address forms the physical 22-bit address (see Figure 14).

A DPR register cannot be modified via an addressing mode that uses the same DPR register. For instance, the instruction “POPW DPR0” is legal only if the stack is kept either in the register file or in a memory location above 8000h, where DPR2 and DPR3 are used. Otherwise, since DPR0 and DPR1 are modified by the instruction, unpredictable behaviour could result.

Figure 14. Addressing via DPR0-3



ADDRESS SPACE EXTENSION (Cont'd)

2.6.2 Addressing 64-Kbyte Segments

This extension mode is used to address Data memory space during a DMA and Program memory space during any code execution (normal code and interrupt routines).

Three registers are used: CSR, ISR, and DMASR. The 6-bit contents of one of the registers CSR, ISR, or DMASR define one out of 64 Memory segments of 64 Kbytes within the 4 Mbytes address space. The registers' contents represent the 6 MSBs of the memory address, whereas the 16 LSBs of the address (intra-segment address) are given by the virtual 16-bit address (seeFigure 15).

2.7 MMU REGISTERS

The MMU uses 7 registers mapped into Group F, Page 21 of the Register File and 2 bits of the EMR2 register (this register is described in the External Memory Interface chapter).

Most of these registers do not have a default value following reset.

2.7.1 DPR0, DPR1, DPR2, DPR3: Data Page Registers

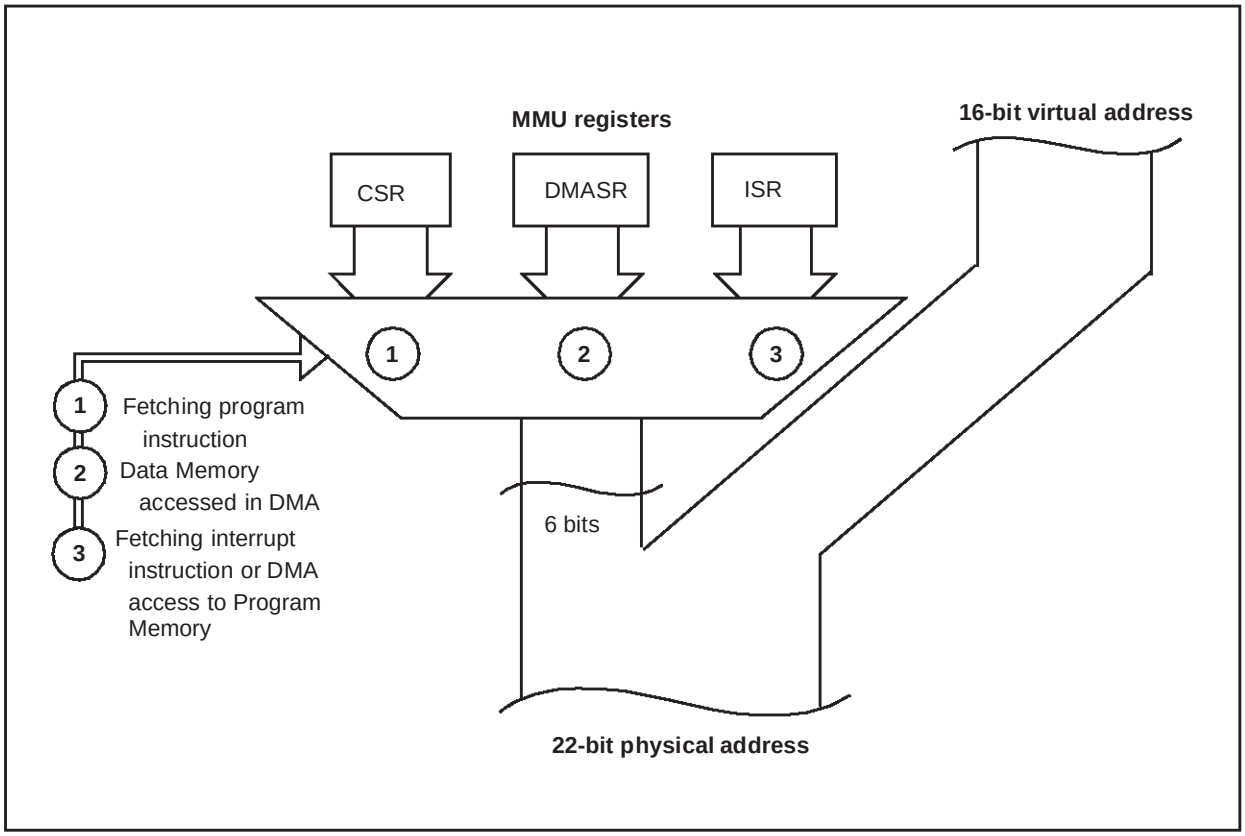
The DPR registers allow access to the entire 4-Mbyte memory space composed of 256 pages of 16 Kbytes.

2.7.1.1 Data Page Register Relocation

If these registers are to be used frequently, they may be relocated in register group E, by programming bit 5 of the EMR2-R246 register in page 21. If this bit is set, the DPR0-3 registers are located at R224-227 in place of the Port 0-3 Data Registers, which are re-mapped to the default DPR's locations: R240-243 page 21.

Data Page Register relocation is illustrated inFigure 13.

Figure 15. Addressing via CSR, ISR, and DMASR



MMU REGISTERS (Cont'd)

DATA PAGE REGISTER 0 (DPR0)
R240 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R224 if EMR2.5 is set.

7							0
DPR0_7	DPR0_6	DPR0_5	DPR0_4	DPR0_3	DPR0_2	DPR0_1	DPR0_0

Bit 7:0 = **DPR0_[7:0]**: These bits define the 16-Kbyte Data Memory page number. They are used as the most significant address bits (A21-14) to extend the address during a Data Memory access. The DPR0 register is used when addressing the virtual address range 0000h-3FFFh.

DATA PAGE REGISTER 1 (DPR1)
R241 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R225 if EMR2.5 is set.

7							0
DPR1_7	DPR1_6	DPR1_5	DPR1_4	DPR1_3	DPR1_2	DPR1_1	DPR1_0

Bit 7:0 = **DPR1_[7:0]**: These bits define the 16-Kbyte Data Memory page number. They are used as the most significant address bits (A21-14) to extend the address during a Data Memory access. The DPR1 register is used when addressing the virtual address range 4000h-7FFFh.

DATA PAGE REGISTER 2 (DPR2)
R242 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R226 if EMR2.5 is set.

7							0
DPR2_7	DPR2_6	DPR2_5	DPR2_4	DPR2_3	DPR2_2	DPR2_1	DPR2_0

Bit 7:0 = **DPR2_[7:0]**: These bits define the 16-Kbyte Data memory page. They are used as the most significant address bits (A21-14) to extend the address during a Data memory access. The DPR2 register is involved when the virtual address is in the range 8000h-BFFFh.

DATA PAGE REGISTER 3 (DPR3)
R243 - Read/Write
Register Page: 21
Reset value: undefined

This register is relocated to R227 if EMR2.5 is set.

7							0
DPR3_7	DPR3_6	DPR3_5	DPR3_4	DPR3_3	DPR3_2	DPR3_1	DPR3_0

Bit 7:0 = **DPR3_[7:0]**: These bits define the 16-Kbyte Data memory page. They are used as the most significant address bits (A21-14) to extend the address during a Data memory access. The DPR3 register is involved when the virtual address is in the range C000h-FFFFh.

MMU REGISTERS (Cont'd)

2.7.2 CSR: Code Segment Register

This register selects the 64-Kbyte code segment being used at run-time to access instructions. It can also be used to access data if the `spm` instruction has been executed (or `ldpp, ldpd, lddp`). Only the 6 LSBs of the CSR register are implemented, and bits 6 and 7 are reserved. The CSR register allows access to the entire memory space, divided into 64 segments of 64 Kbytes.

To generate the 22-bit Program memory address, the contents of the CSR register is directly used as the 6 MSBs, and the 16-bit virtual address as the 16 LSBs.

Note: The CSR register should only be read and not written for data operations (there are some exceptions which are documented in the following paragraph). It is, however, modified either directly by means of the `jps` and `calls` instructions, or indirectly via the stack, by means of the `rets` instruction.

CODE SEGMENT REGISTER (CSR)
R244 **-** Read/Write
Register Page: 21
Reset value: 0000 0000 (00h)

7							0
0	0	CSR_5	CSR_4	CSR_3	CSR_2	CSR_1	CSR_0

Bit 7:6 = Reserved, keep in reset state.

Bit 5:0 = **CSR_[5:0]**: These bits define the 64-Kbyte memory segment (among 64) which contains the code being executed. These bits are used as the most significant address bits (A21-16).

2.7.3 ISR: Interrupt Segment Register

INTERRUPT SEGMENT REGISTER (ISR)
R248 - Read/Write
Register Page: 21
Reset value: undefined

7							0
0	0	ISR_5	ISR_4	ISR_3	ISR_2	ISR_1	ISR_0

ISR and ENCSR bit (EMR2 register) are also described in the chapter relating to Interrupts, please refer to this description for further details.

Bit 7:6 = Reserved, keep in reset state.

Bit 5:0 = **ISR_[5:0]**: These bits define the 64-Kbyte memory segment (among 64) which contains the interrupt vector table and the code for interrupt service routines and DMA transfers (when the PS bit of the DAPR register is reset). These bits are used as the most significant address bits (A21-16). The ISR is used to extend the address space in two cases:

– Whenever an interrupt occurs: ISR points to the 64-Kbyte memory segment containing the interrupt vector table and the interrupt service routine code. See also the Interrupts chapter.

– During DMA transactions between the peripheral and memory when the PS bit of the DAPR register is reset : ISR points to the 64 K-byte Memory segment that will be involved in the DMA transaction.

2.7.4 DMASR: DMA Segment Register

DMA SEGMENT REGISTER (DMASR)
R249 - Read/Write
Register Page: 21
Reset value: undefined

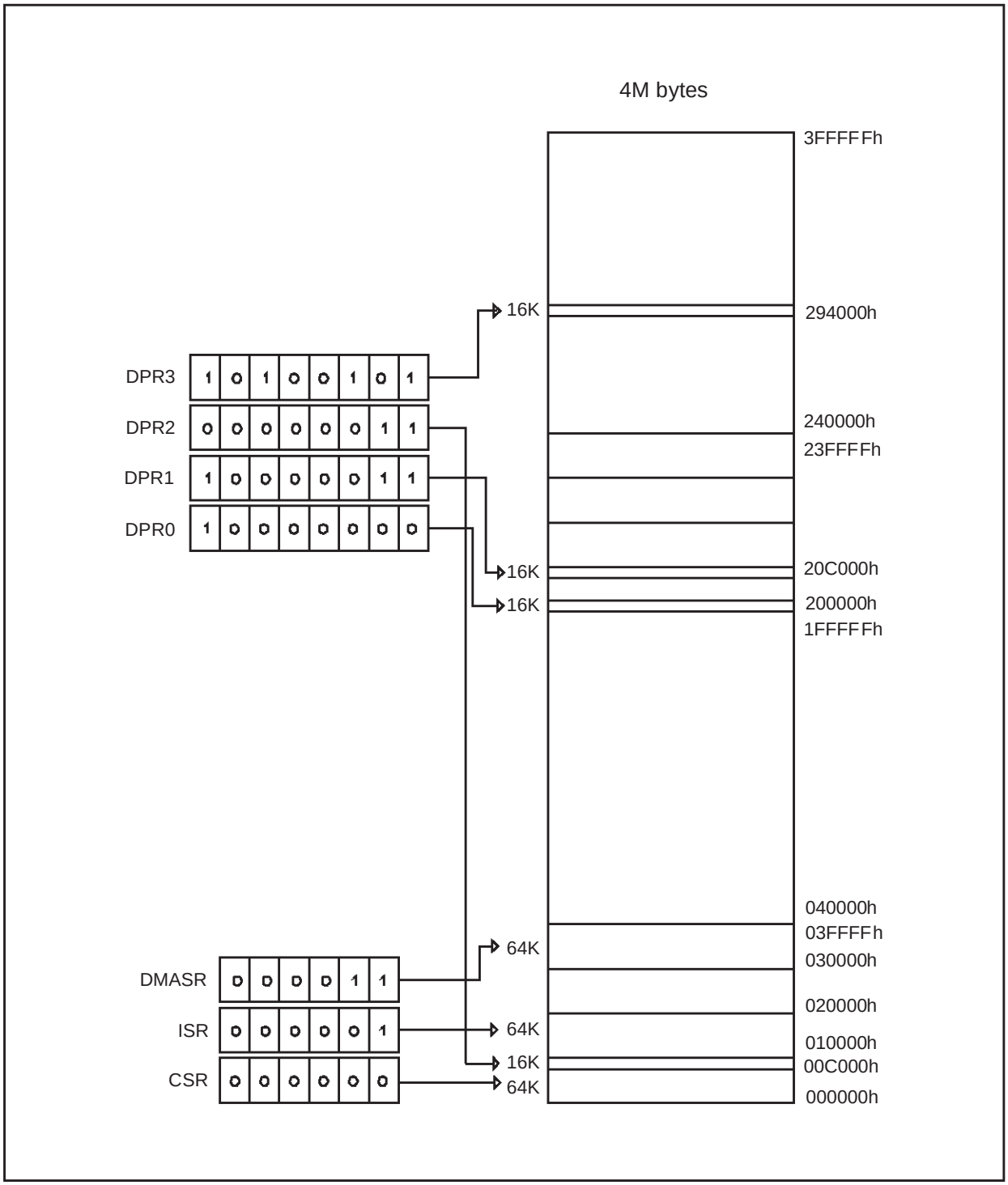
7							0
0	0	DMA SR_5	DMA SR_4	DMA SR_3	DMA SR_2	DMA SR_1	DMA SR_0

Bit 7:6 = Reserved, keep in reset state.

Bit 5:0 = **DMASR_[5:0]**: These bits define the 64-Kbyte Memory segment (among 64) used when a DMA transaction is performed between the peripheral's data register and Memory, with the PS bit of the DAPR register set. These bits are used as the most significant address bits (A21-16). If the PS bit is reset, the ISR register is used to extend the address.

MMU REGISTERS (Cont'd)

Figure 16. Memory Addressing Scheme (example)



2.8 MMU USAGE

2.8.1 Normal Program Execution

Program memory is organized as a set of 64-Kbyte segments. The program can span as many segments as needed, but a procedure cannot stretch across segment boundaries. `jps`, `calls` and `rets` instructions, which automatically modify the CSR, must be used to jump across segment boundaries. Writing to the CSR is forbidden during normal program execution because it is not synchronized with the opcode fetch. This could result in fetching the first byte of an instruction from one memory segment and the second byte from another. Writing to the CSR is allowed when it is not being used, i.e during an interrupt service routine if ENCSR is reset.

Note that a routine must always be called in the same way, i.e. either always with `call` or always with `calls`, depending on whether the routine ends with `ret` or `rets`. This means that if the routine is written without prior knowledge of the location of other routines which call it, and all the program code does not fit into a single 64-Kbyte segment, then `calls`/`rets` should be used.

In typical microcontroller applications, less than 64 Kbytes of RAM are used, so the four Data space pages are normally sufficient, and no change of DPR0-3 is needed during Program execution. It may be useful however to map part of the ROM into the data space if it contains strings, tables, bit maps, etc.

If there is to be frequent use of paging, the user can set bit 5 (DPRREM) in register R246 (EMR2) of Page 21. This swaps the location of registers DPR0-3 with that of the data registers of Ports 0-3. In this way, DPR registers can be accessed without the need to save/set/restore the Page Pointer Register. Port registers are therefore moved to page 21. Applications that require a lot of paging typically use more than 64 Kbytes of external memory, and as ports 0, 1 and 2 are required to address it, their data registers are unused.

MMU USAGE (Cont'd)

2.8.2 Interrupts

The ISR register has been created so that the interrupt routines may be found by means of the same vector table even after a segment jump/call.

When an interrupt occurs, the CPU behaves in one of 2 ways, depending on the value of the ENCSR bit in the EMR2 register (R246 on Page 21).

If this bit is reset (default condition), the CPU works in original ST9 compatibility mode. For the duration of the interrupt service routine, the ISR is used instead of the CSR, and the interrupt stack frame is kept exactly as in the original ST9 (only the PC and flags are pushed). This avoids the need to save the CSR on the stack in the case of an interrupt, ensuring a fast interrupt response time. The drawback is that it is not possible for an interrupt service routine to perform segment `calls`/`jps`: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code size of all interrupt service routines is thus limited to 64 Kbytes.

If, instead, bit 6 of the EMR2 register is set, the ISR is used only to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and the flags, and then the CSR is loaded with the ISR. In this case, an `iret` will also restore the CSR from the stack. This approach lets interrupt service routines access the whole 4-Mbyte address space. The drawback is that the interrupt response time is slightly increased, because of the need to also save the CSR on the stack. Compatibility with the original ST9 is also lost in this case, because the interrupt stack frame is different; this difference, however, would not be noticeable for a vast majority of programs.

Data memory mapping is independent of the value of bit 6 of the EMR2 register, and remains the same as for normal code execution: the stack is the same as that used by the main program, as in the ST9. If the interrupt service routine needs to access additional Data memory, it must save one (or more) of the DPRs, load it with the needed memory page and restore it before completion.

2.8.3 DMA

Depending on the PS bit in the DAPR register (see DMA chapter) DMA uses either the ISR or the DMASR for memory accesses: this guarantees that a DMA will always find its memory segment(s), no matter what segment changes the application has performed. Unlike interrupts, DMA transactions cannot save/restore paging registers, so a dedicated segment register (DMASR) has been created. Having only one register of this kind means that all DMA accesses should be programmed in one of the two following segments: the one pointed to by the ISR (when the PS bit of the DAPR register is reset), and the one referenced by the DMASR (when the PS bit is set).

3 REGISTER AND MEMORY MAP

3.1 MEMORY CONFIGURATION

The Program memory space of the ST90135/158, 0/16/24/32/48/64/K bytes of directly addressable on-chip memory, is fully available to the user.

The first 256 memory locations from address 0 to FFh hold the Reset Vector, the Top-Level (Pseudo Non-Maskable) interrupt, the Divide by Zero Trap Routine vector and, optionally, the interrupt vector table for use with the on-chip peripherals and the external interrupt sources. Apart from this case no other part of the Program memory has a predetermined function except segment 21h which is reserved for use by STMicroelectronics.

3.2 EPROM PROGRAMMING

The 65536 bytes of EPROM memory of the ST90E158 may be programmed by using the EPROM Programming Boards (EPB) or gang programmers available from STMicroelectronics.

EPROM Erasing

The EPROM of the windowed package of the ST90E158 may be erased by exposure to Ultra-Violet light.

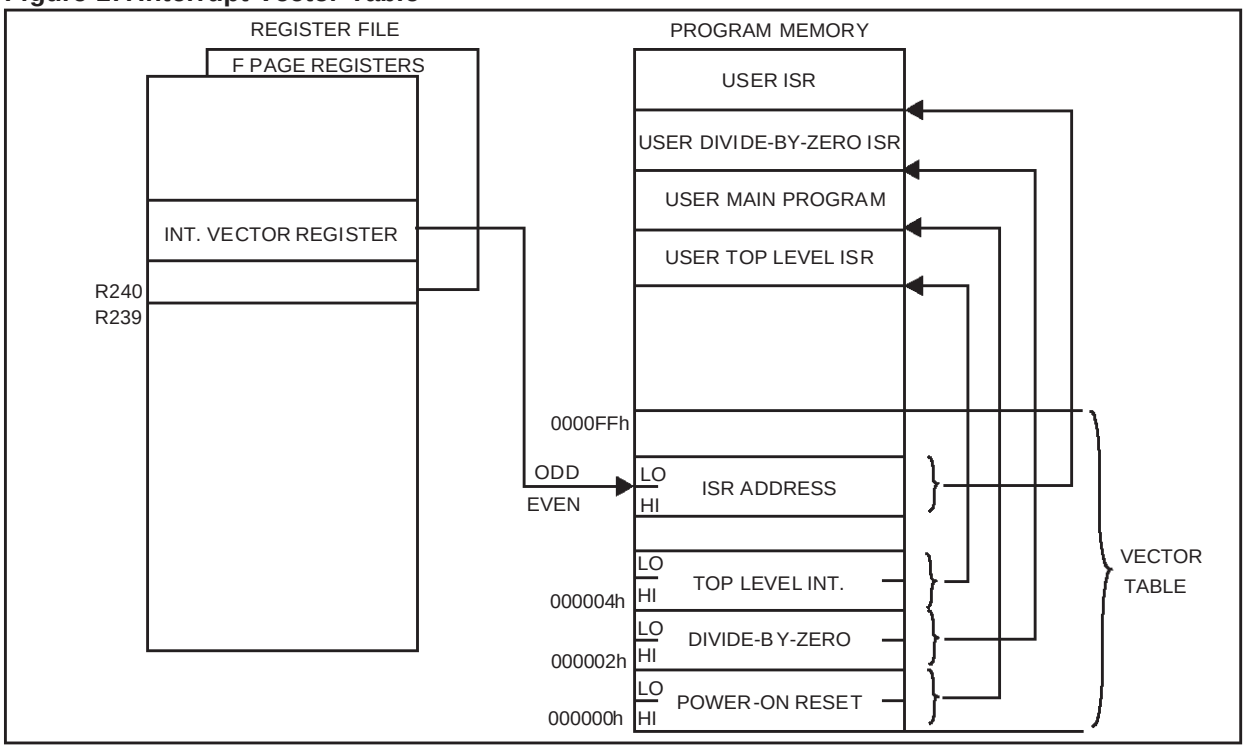
The erasure characteristic of the ST90E158 is such that erasure begins when the memory is exposed to light with a wave lengths shorter than approximately 4000Å. It should be noted that sunlight and some types of fluorescent lamps have wavelengths in the range 3000-4000Å. It is thus recommended that the window of the ST90E158 packages be covered by an opaque label to prevent unintentional erasure problems when testing the application in such an environment.

The recommended erasure procedure of the EPROM is the exposure to short wave ultraviolet light which have a wave-length 2537Å. The integrated dose (i.e. U.V. intensity x exposure time) for erasure should be a minimum of 15W-sec/cm2. The erasure time with this dosage is approximately 30 minutes using an ultraviolet lamp with 12000mW/cm2 power rating. The ST90E158 should be placed within 2.5cm (1 inch) of the lamp tubes during erasure.

Table 6. First 6 Bytes of Program Space

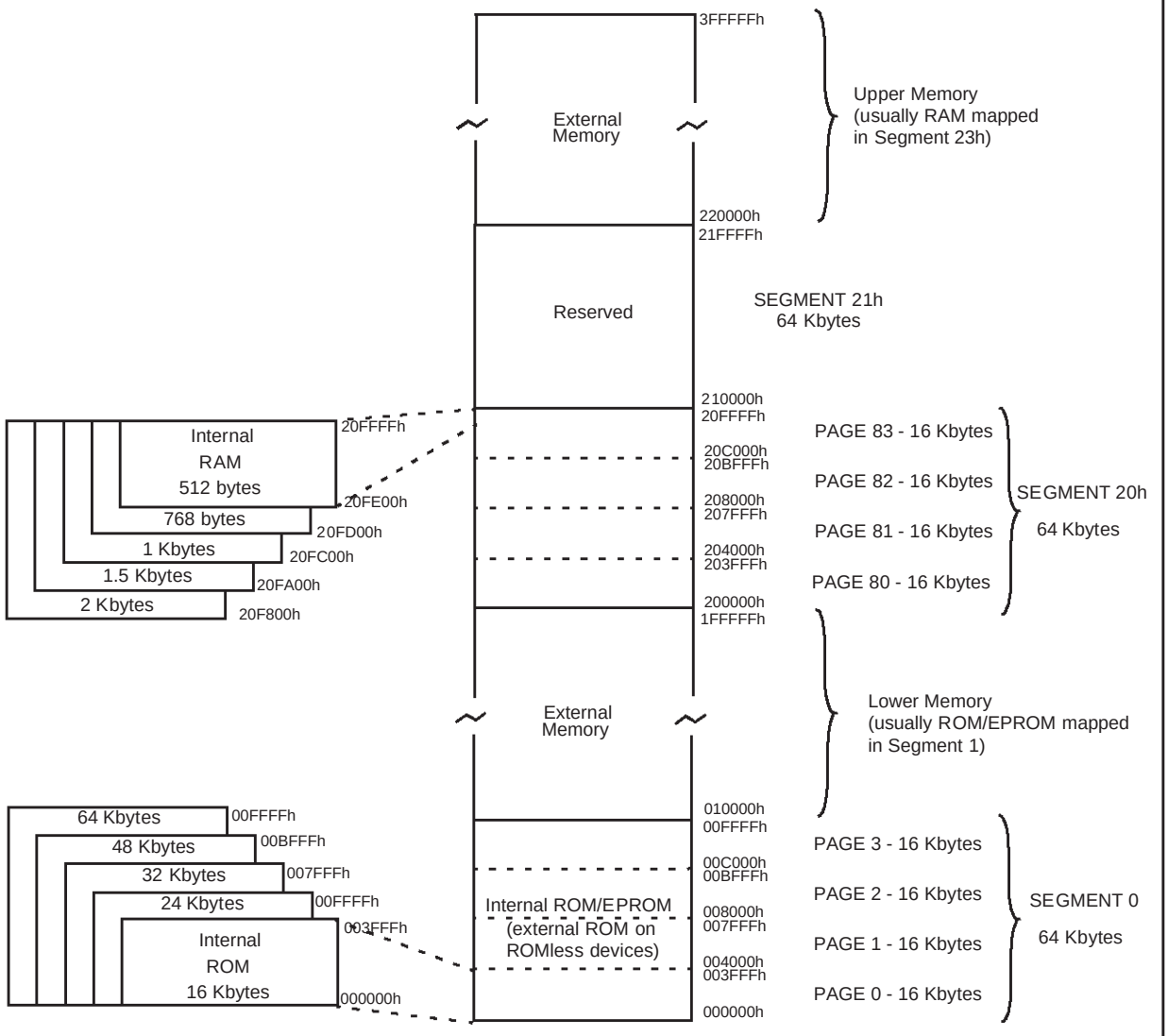
0	Address high of Power on Reset routine
1	Address low of Power on Reset routine
2	Address high of Divide by zero trap Subroutine
3	Address low of Divide by zero trap Subroutine
4	Address high of Top Level Interrupt routine
5	Address low of Top Level Interrupt routine

Figure 17. Interrupt Vector Table



3.3 MEMORY MAP

Figure 18. Memory Map



Note: The total amount of directly addressable external memory is 64 Kbytes.

3.4 ST90158/135 REGISTER MAP

The following pages contain a list of ST90158/135 registers, grouped by peripheral or function.

Be very careful to correctly program both:

– The set of registers dedicated to a particular function or peripheral.

– Registers common to other functions.

– In particular, double-check that any registers with “undefined” reset values have been correctly initialised.

Warning: Note that in the **EIVR** and each **IVR** register, all bits are significant. Take care when defining base vector addresses that entries in the Interrupt Vector table do not overlap.

Table 7. Common Registers

Function or Peripheral	Common Registers
SCI, MFT	CICR + NICR + DMA REGISTERS + I/O PORT REGISTERS
ADC	CICR + NICR + I/O PORT REGISTERS
SPI, WDT, STIM	CICR + NICR + EXTERNAL INTERRUPT REGISTERS + I/O PORT REGISTERS
I/O PORTS	I/O PORT REGISTERS + MODER
EXTERNAL INTERRUPT	INTERRUPT REGISTERS + I/O PORT REGISTERS
RCCU	INTERRUPT REGISTERS + MODER

Table 8. Group F Pages

Resources available on the ST90158/ST90135 devices:

Register	Page													
	0	2	3	8	9	10	11	12	13	21	24	25	43	55	63
R255	Res.	Res.	PORT 7	MFT1	Res.	MFT0 (*)	Res.	MFT3	Res.	Res.	SCI0	SCI1 (*)	PORT 9	Res.	A/D
R254	SPI														
R253															
R252	WCR														
R251	WDT		PORT 6										PORT 8		
R250		PORT 2								MMU					
R249															
R248					MFT										
R247	EXT INT	Res.	Res.		MFT1				MFT3	Res.			Res.	RCCU	
R246		PORT 1	PORT 5							EXT MI					
R245														Res.	
R244										MMU					
R243		Res.	Res.		MFT0 (*)		STIM		Res.						
R242		PORT 0	PORT 4											RCCU	
R241	Res.													Res.	
R240														RCCU	

(*) ST90158/ST90E158 only. Not present on ST90135.

Table 9. Detailed Register Map

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
N/A	Core	R230	CICR	Central Interrupt Control Register	87
		R231	FLAGR	Flag Register	00
		R232	RP0	Pointer 0 Register	xx
		R233	RP1	Pointer 1 Register	xx
		R234	PPR	Page Pointer Register	xx
		R235	MODER	Mode Register	E0
		R236	USPHR	User Stack Pointer High Register	xx
		R237	USPLR	User Stack Pointer Low Register	xx
		R238	SSPHR	System Stack Pointer High Reg.	xx
		R239	SSPLR	System Stack Pointer Low Reg.	xx
	I/O Port 5:4,2:0	R224	P0DR	Port 0 Data Register	FF
		R225	P1DR	Port 1 Data Register	FF
		R226	P2DR	Port 2 Data Register	FF
		R228	P4DR	Port 4 Data Register	FF
		R229	P5DR	Port 5 Data Register	FF
0	INT	R242	EITR	External Interrupt Trigger Register	00
		R243	EIPR	External Interrupt Pending Reg.	00
		R244	EIMR	External Interrupt Mask-bit Reg.	00
		R245	EIPLR	External Interrupt Priority Level Reg.	FF
		R246	EIVR	External Interrupt Vector Register	x6
		R247	NICR	Nested Interrupt Control	00
	WDT	R248	WDTHR	Watchdog Timer High Register	FF
		R249	WDTLR	Watchdog Timer Low Register	FF
		R250	WDTPR	Watchdog Timer Prescaler Reg.	FF
		R251	WDTCR	Watchdog Timer Control Register	12
		R252	WCR	Wait Control Register	7F
	SPI	R253	SPIDR	SPI Data Register	xx
		R254	SPICR	SPI Control Register	00
2	I/O Port 0	R240	P0C0	Port 0 Configuration Register 0	00
		R241	P0C1	Port 0 Configuration Register 1	00
		R242	P0C2	Port 0 Configuration Register 2	00
	I/O Port 1	R244	P1C0	Port 1 Configuration Register 0	00
		R245	P1C1	Port 1 Configuration Register 1	00
		R246	P1C2	Port 1 Configuration Register 2	00
	I/O Port 2	R248	P2C0	Port 2 Configuration Register 0	FF
		R249	P2C1	Port 2 Configuration Register 1	00
		R250	P2C2	Port 2 Configuration Register 2	00

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
3	I/O Port 4	R240	P4C0	Port 4 Configuration Register 0	FF
		R241	P4C1	Port 4 Configuration Register 1	00
		R242	P4C2	Port 4 Configuration Register 2	00
	I/O Port 5	R244	P5C0	Port 5 Configuration Register 0	FF
		R245	P5C1	Port 5 Configuration Register 1	00
		R246	P5C2	Port 5 Configuration Register 2	00
	I/O Port 6	R248	P6C0	Port 6 Configuration Register 0	FF
		R249	P6C1	Port 6 Configuration Register 1	00
		R250	P6C2	Port 6 Configuration Register 2	00
		R251	P6DR	Port 6 Data Register	FF
	I/O Port 7	R252	P7C0	Port 7 Configuration Register 0	00/FF
		R253	P7C1	Port 7 Configuration Register 1	00/00
		R254	P7C2	Port 7 Configuration Register 2	00/00
		R255	P7DR	Port 7 Data Register	FF

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
8	MFT1	R240	REG0HR1	Capture Load Register 0 High	xx
		R241	REG0LR1	Capture Load Register 0 Low	xx
		R242	REG1HR1	Capture Load Register 1 High	xx
		R243	REG1LR1	Capture Load Register 1 Low	xx
		R244	CMP0HR1	Compare 0 Register High	00
		R245	CMP0LR1	Compare 0 Register Low	00
		R246	CMP1HR1	Compare 1 Register High	00
		R247	CMP1LR1	Compare 1 Register Low	00
		R248	TCR1	Timer Control Register	0x
		R249	TMR1	Timer Mode Register	00
		R250	ICR1	External Input Control Register	0x
		R251	PRSR1	Prescaler Register	00
		R252	OACR1	Output A Control Register	xx
		R253	OBCR1	Output B Control Register	xx
		R254	FLAGR1	Flags Register	00
		R255	IDMR1	Interrupt/DMA Mask Register	00
9		R244	DCPR0	DMA Counter Pointer Register	xx
		R245	DAPR0	DMA Address Pointer Register	xx
		R246	IVR0	Interrupt Vector Register	xx
		R247	IDCR0	Interrupt/DMA Control Register	C7
	MFT0,1	R248	IOCR	I/O Connection Register	FC
	MFT0 (*)	R240	DCPR1	DMA Counter Pointer Register	xx
		R241	DAPR1	DMA Address Pointer Register	xx
		R242	IVR1	Interrupt Vector Register	xx
		R243	IDCR1	Interrupt/DMA Control Register	C7
10		R240	REG0HR0	Capture Load Register 0 High	xx
		R241	REG0LR0	Capture Load Register 0 Low	xx
		R242	REG1HR0	Capture Load Register 1 High	xx
		R243	REG1LR0	Capture Load Register 1 Low	xx
		R244	CMP0HR0	Compare 0 Register High	00
		R245	CMP0LR0	Compare 0 Register Low	00
		R246	CMP1HR0	Compare 1 Register High	00
		R247	CMP1LR0	Compare 1 Register Low	00
		R248	TCR0	Timer Control Register	0x
		R249	TMR0	Timer Mode Register	00
		R250	ICR0	External Input Control Register	0x
		R251	PRSR0	Prescaler Register	00
		R252	OACR0	Output A Control Register	xx
		R253	OBCR0	Output B Control Register	xx
		R254	FLAGR0	Flags Register	00
		R255	IDMR0	Interrupt/DMA Mask Register	00

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
11	STIM	R240	STH	Counter High Byte Register	FF
		R241	STL	Counter Low Byte Register	FF
		R242	STP	Standard Timer Prescaler Register	FF
		R243	STC	Standard Timer Control Register	14
12	MFT3	R240	REG0HR1	Capture Load Register 0 High	xx
		R241	REG0LR1	Capture Load Register 0 Low	xx
		R242	REG1HR1	Capture Load Register 1 High	xx
		R243	REG1LR1	Capture Load Register 1 Low	xx
		R244	CMP0HR1	Compare 0 Register High	00
		R245	CMP0LR1	Compare 0 Register Low	00
		R246	CMP1HR1	Compare 1 Register High	00
		R247	CMP1LR1	Compare 1 Register Low	00
		R248	TCR1	Timer Control Register	0x
		R249	TMR1	Timer Mode Register	00
		R250	ICR1	External Input Control Register	0x
		R251	PRSR1	Prescaler Register	00
		R252	OACR1	Output A Control Register	xx
		R253	OBCR1	Output B Control Register	xx
		R254	FLAGR1	Flags Register	00
		R255	IDMR1	Interrupt/DMA Mask Register	00
13		R244	DCPR0	DMA Counter Pointer Register	xx
		R245	DAPR0	DMA Address Pointer Register	xx
		R246	IVR0	Interrupt Vector Register	xx
		R247	IDCR0	Interrupt/DMA Control Register	C7
21	MMU	R240	DPR0	Data Page Register 0	xx
		R241	DPR1	Data Page Register 1	xx
		R242	DPR2	Data Page Register 2	xx
		R243	DPR3	Data Page Register 3	xx
		R244	CSR	Code Segment Register	00
		R248	ISR	Interrupt Segment Register	xx
		R249	DMASR	DMA Segment Register	xx
	EXTMI	R245	EMR1	External Memory Register 1	80
		R246	EMR2	External Memory Register 2	0F

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
24	SCI0	R240	RDCPR0	Receiver DMA Transaction Counter Pointer	xx
		R241	RDAPR0	Receiver DMA Source Address Pointer	xx
		R242	TDCPR0	Transmitter DMA Transaction Counter Pointer	xx
		R243	TDAPR0	Transmitter DMA Destination Address Pointer	xx
		R244	IVR0	Interrupt Vector Register	xx
		R245	ACR0	Address/Data Compare Register	xx
		R246	IMR0	Interrupt Mask Register	x0
		R247	ISR0	Interrupt Status Register	xx
		R248	RXBR0	Receive Buffer Register	xx
		R248	TXBR0	Transmitter Buffer Register	xx
		R249	IDPR0	Interrupt/DMA Priority Register	xx
		R250	CHCR0	Character Configuration Register	xx
		R251	CCR0	Clock Configuration Register	00
		R252	BRGHR0	Baud Rate Generator High Reg.	xx
		R253	BRGLR0	Baud Rate Generator Low Register	xx
		R254	SICR0	Synchronous Input Control	03
		R255	SOCR0	Synchronous Output Control	01
25	SCI1 (*)	R240	RDCPR1	Receiver DMA Transaction Counter Pointer	xx
		R241	RDAPR1	Receiver DMA Source Address Pointer	xx
		R242	TDCPR1	Transmitter DMA Transaction Counter Pointer	xx
		R243	TDAPR1	Transmitter DMA Destination Address Pointer	xx
		R244	IVR1	Interrupt Vector Register	xx
		R245	ACR1	Address/Data Compare Register	xx
		R246	IMR1	Interrupt Mask Register	x0
		R247	ISR1	Interrupt Status Register	xx
		R248	RXBR1	Receive Buffer Register	xx
		R248	TXBR1	Transmitter Buffer Register	xx
		R249	IDPR1	Interrupt/DMA Priority Register	xx
		R250	CHCR1	Character Configuration Register	xx
		R251	CCR1	Clock Configuration Register	00
		R252	BRGHR1	Baud Rate Generator High Reg.	xx
		R253	BRGLR1	Baud Rate Generator Low Register	xx
		R254	SICR1	Synchronous Input Control	03
		R255	SOCR1	Synchronous Output Control	01
43	I/O Port 8	R248	P8C0	Port 8 Configuration Register 0	00/03
		R249	P8C1	Port 8 Configuration Register 1	00/00
		R250	P8C2	Port 8 Configuration Register 2	00/00
		R251	P8DR	Port 8 Data Register	FF
	I/O Port 9	R252	P9C0	Port 9 Configuration Register 0	00/00
		R253	P9C1	Port 9 Configuration Register 1	00/00
		R254	P9C2	Port 9 Configuration Register 2	00/00
		R255	P9DR	Port 9 Data Register	FF

Page (Decimal)	Block	Reg. No.	Register Name	Description	Reset Value Hex.
55	RCCU	R240	CLKCTL	Clock Control Register	00
		R242	CLK_FLAG	Clock Flag Register	48, 28 or 08
		R246	PLLCONF	PLL Configuration Register	xx
63	AD0	R240	D0R0	Channel 0 Data Register	xx
		R241	D1R0	Channel 1 Data Register	xx
		R242	D2R0	Channel 2 Data Register	xx
		R243	D3R0	Channel 3 Data Register	xx
		R244	D4R0	Channel 4 Data Register	xx
		R245	D5R0	Channel 5 Data Register	xx
		R246	D6R0	Channel 6 Data Register	xx
		R247	D7R0	Channel 7 Data Register	xx
		R248	LT6R0	Channel 6 Lower Threshold Reg.	xx
		R249	LT7R0	Channel 7 Lower Threshold Reg.	xx
		R250	UT6R0	Channel 6 Upper Threshold Reg.	xx
		R251	UT7R0	Channel 7 Upper Threshold Reg.	xx
		R252	CRR0	Compare Result Register	0F
		R253	CLR0	Control Logic Register	00
		R254	ICR0	Interrupt Control Register	0F
		R255	IVR0	Interrupt Vector Register	x2

(*) Not present on ST90135.

Note: xx denotes a byte with an undefined value, however some of the bits may have defined values. Refer to register description for details.

4 INTERRUPTS

4.1 INTRODUCTION

The ST9 responds to peripheral and external events through its interrupt channels. Current program execution can be suspended to allow the ST9 to execute a specific response routine when such an event occurs, providing that interrupts have been enabled, and according to a priority mechanism. If an event generates a valid interrupt request, the current program status is saved and control passes to the appropriate Interrupt Service Routine.

The ST9 CPU can receive requests from the following sources:

– On-chip peripherals

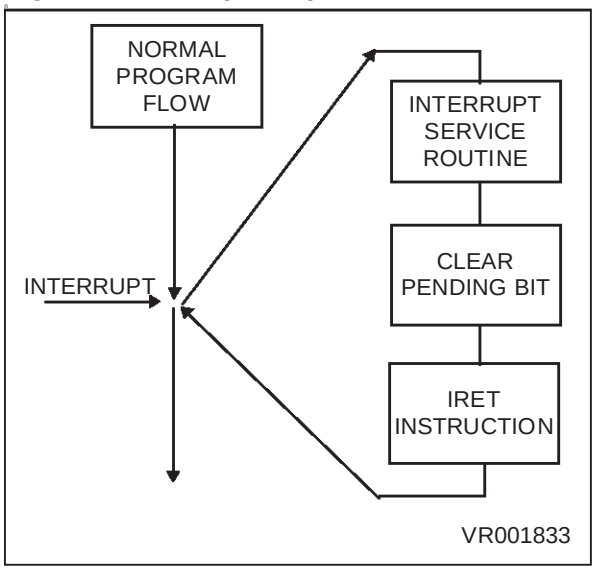
– External pins

– Top-Level Pseudo-non-maskable interrupt

According to the on-chip peripheral features, an event occurrence can generate an Interrupt request which depends on the selected mode.

Up to eight external interrupt channels, with programmable input trigger edge, are available. In addition, a dedicated interrupt channel, set to the Top-level priority, can be devoted either to the external NMI pin (where available) to provide a Non-Maskable Interrupt, or to the Timer/Watchdog. Interrupt service routines are addressed through a vector table mapped in Memory.

Figure 19. Interrupt Response



4.2 INTERRUPT VECTORING

The ST9 implements an interrupt vectoring structure which allows the on-chip peripheral to identify the location of the first instruction of the Interrupt Service Routine automatically.

When an interrupt request is acknowledged, the peripheral interrupt module provides, through its Interrupt Vector Register (IVR), a vector to point into the vector table of locations containing the start addresses of the Interrupt Service Routines (defined by the programmer).

Each peripheral has a specific IVR mapped within its Register File pages.

The Interrupt Vector table, containing the addresses of the Interrupt Service Routines, is located in the first 256 locations of Memory pointed to by the ISR register, thus allowing 8-bit vector addressing. For a description of the ISR register refer to the chapter describing the MMU.

The user Power on Reset vector is stored in the first two physical bytes in memory, 000000h and 000001h.

The Top Level Interrupt vector is located at addresses 0004h and 0005h in the segment pointed to by the Interrupt Segment Register (ISR).

With one Interrupt Vector register, it is possible to address several interrupt service routines; in fact, peripherals can share the same interrupt vector register among several interrupt channels. The most significant bits of the vector are user programmable to define the base vector address within the vector table, the least significant bits are controlled by the interrupt module, in hardware, to select the appropriate vector.

Note: The first 256 locations of the memory segment pointed to by ISR can contain program code.

4.2.1 Divide by Zero trap

The Divide by Zero trap vector is located at addresses 0002h and 0003h of each code segment; it should be noted that for each code segment a Divide by Zero service routine is required.

Warning. *Although the Divide by Zero Trap operates as an interrupt, the FLAG Register is not pushed onto the system Stack automatically. As a result it must be regarded as a subroutine, and the service routine must end with the* `RET` *instruction (not* `IRET` *).*

4.2.2 Segment Paging During Interrupt Routines

The ENCSR bit in the EMR2 register can be used to select between original ST9 backward compatibility mode and ST9+ interrupt management mode.

ST9 backward compatibility mode(ENCSR = 0)

If ENCSR is reset, the CPU works in original ST9 compatibility mode. For the duration of the interrupt service routine, ISR is used instead of CSR, and the interrupt stack frame is identical to that of the original ST9: only the PC and Flags are pushed.

This avoids saving the CSR on the stack in the event of an interrupt, thus ensuring a faster interrupt response time.

It is not possible for an interrupt service routine to perform inter-segment calls or jumps: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code segment size for all interrupt service routines is thus limited to 64K bytes.

ST9+ mode (ENCSR = 1)

If ENCSR is set, ISR is only used to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and flags, and CSR is then loaded with the contents of ISR.

In this case, `iret` will also restore CSR from the stack. This approach allows interrupt service routines to access the entire 4 Mbytes of address space. The drawback is that the interrupt response time is slightly increased, because of the need to also save CSR on the stack.

Full compatibility with the original ST9 is lost in this case, because the interrupt stack frame is different.

ENCSR Bit	0	1
Mode	ST9 Compatible	ST9+
Pushed/Popped Registers	PC, FLAGR	PC, FLAGR, CSR
Max. Code Size for interrupt service routine	64KB Within 1 segment	No limit Across segments

4.3 INTERRUPT PRIORITY LEVELS

The ST9 supports a fully programmable interrupt priority structure. Nine priority levels are available to define the channel priority relationships:

– The on-chip peripheral channels and the eight external interrupt sources can be programmed within eight priority levels. Each channel has a 3-bit field, PRL (Priority Level), that defines its priority level in the range from 0 (highest priority) to 7 (lowest priority).

– The 9th level (Top Level Priority) is reserved for the Timer/Watchdog or the External Pseudo Non-Maskable Interrupt. An Interrupt service routine at this level cannot be interrupted in any arbitration mode. Its mask can be both maskable (TLI) or non-maskable (TLNM).

4.4 PRIORITY LEVEL ARBITRATION

The 3 bits of CPL (Current Priority Level) in the Central Interrupt Control Register contain the priority of the currently running program (CPU priority). CPL is set to 7 (lowest priority) upon reset and can be modified during program execution either by software or automatically by hardware according to the selected Arbitration Mode.

During every instruction, an arbitration phase takes place, during which, for every channel capable of generating an Interrupt, each priority level is compared to all the other requests (interrupts or DMA).

If the highest priority request is an interrupt, its PRL value must be strictly lower (that is, higher priority) than the CPL value stored in the CICR register (R230) in order to be acknowledged. The Top Level Interrupt overrides every other priority.

4.4.1 Priority level 7 (Lowest)

Interrupt requests at PRL level 7 cannot be acknowledged, as this PRL value (the lowest possible priority) cannot be strictly lower than the CPL value. This can be of use in a fully polled interrupt environment.

4.4.2 Maximum depth of nesting

No more than 8 routines can be nested. If an interrupt routine at level N is being serviced, no other Interrupts located at level N can interrupt it. This guarantees a maximum number of 8 nested levels including the Top Level Interrupt request.

4.4.3 Simultaneous Interrupts

If two or more requests occur at the same time and at the same priority level, an on-chip daisy chain, specific to every ST9 version, selects the channel

with the highest position in the chain, as shown in Table 1.

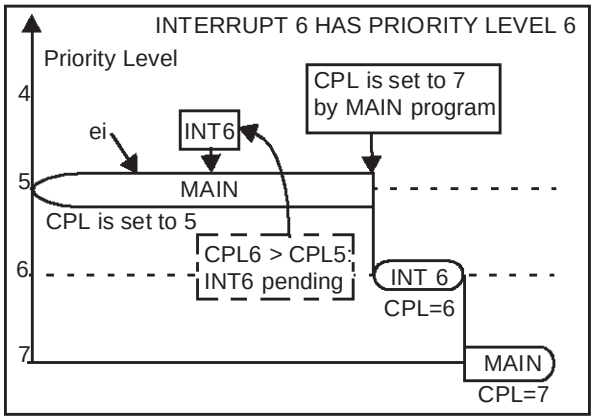
Table 10. Daisy Chain Priority

Highest Position	INTA0	INT0/WDT
	INTA1	INT1
	INTB0	INT2/SPI
	INTB1	INT3
	INTC0	INT4/STIM
	INTC1	INT5
	INTD0	INT6/RCCU
	INTD1	INT7
	TIMER0	
	SCI0	
	SCI1	
	A/D	
	TIMER3	
Lowest Position	TIMER1	

4.4.4 Dynamic Priority Level Modification

The main program and routines can be specifically prioritized. Since the CPL is represented by 3 bits in a read/write register, it is possible to modify dynamically the current priority value during program execution. This means that a critical section can have a higher priority with respect to other interrupt requests. Furthermore it is possible to prioritize even the Main Program execution by modifying the CPL during its execution. See Figure 2

Figure 20. Example of Dynamic priority level modification in Nested Mode



4.5 ARBITRATION MODES

The ST9 provides two interrupt arbitration modes: Concurrent mode and Nested mode. Concurrent mode is the standard interrupt arbitration mode. Nested mode improves the effective interrupt response time when service routine nesting is required, depending on the request priority levels.

The IAM control bit in the CICR Register selects Concurrent Arbitration mode or Nested Arbitration Mode.

4.5.1 Concurrent Mode

This mode is selected when the IAM bit is cleared (reset condition). The arbitration phase, performed during every instruction, selects the request with the highest priority level. The CPL value is not modified in this mode.

Start of Interrupt Routine

The interrupt cycle performs the following steps:

– All maskable interrupt requests are disabled by clearing CICR.IEN.

– The PC low byte is pushed onto system stack.

– The PC high byte is pushed onto system stack.

– If ENCSR is set, CSR is pushed onto system stack.

– The Flag register is pushed onto system stack.

– The PC is loaded with the 16-bit vector stored in the Vector Table, pointed to by the IVR.

– If ENCSR is set, CSR is loaded with ISR contents; otherwise ISR is used in place of CSR until `iret` instruction.

End of Interrupt Routine

The Interrupt Service Routine must be ended with the `iret` instruction. The `iret` instruction executes the following operations:

– The Flag register is popped from system stack.

– If ENCSR is set, CSR is popped from system stack.

– The PC high byte is popped from system stack.

– The PC low byte is popped from system stack.

– All unmasked Interrupts are enabled by setting the CICR.IEN bit.

– If ENCSR is reset, CSR is used instead of ISR.

Normal program execution thus resumes at the interrupted instruction. All pending interrupts remain pending until the next `ei` instruction (even if it is executed during the interrupt service routine).

Note: In Concurrent mode, the source priority level is only useful during the arbitration phase, where it is compared with all other priority levels and with the CPL. No trace is kept of its value during the ISR. If other requests are issued during the interrupt service routine, once the global CICR.IEN is re-enabled, they will be acknowledged regardless of the interrupt service routine's priority. This may cause undesirable interrupt response sequences.

ARBITRATION MODES (Cont'd)

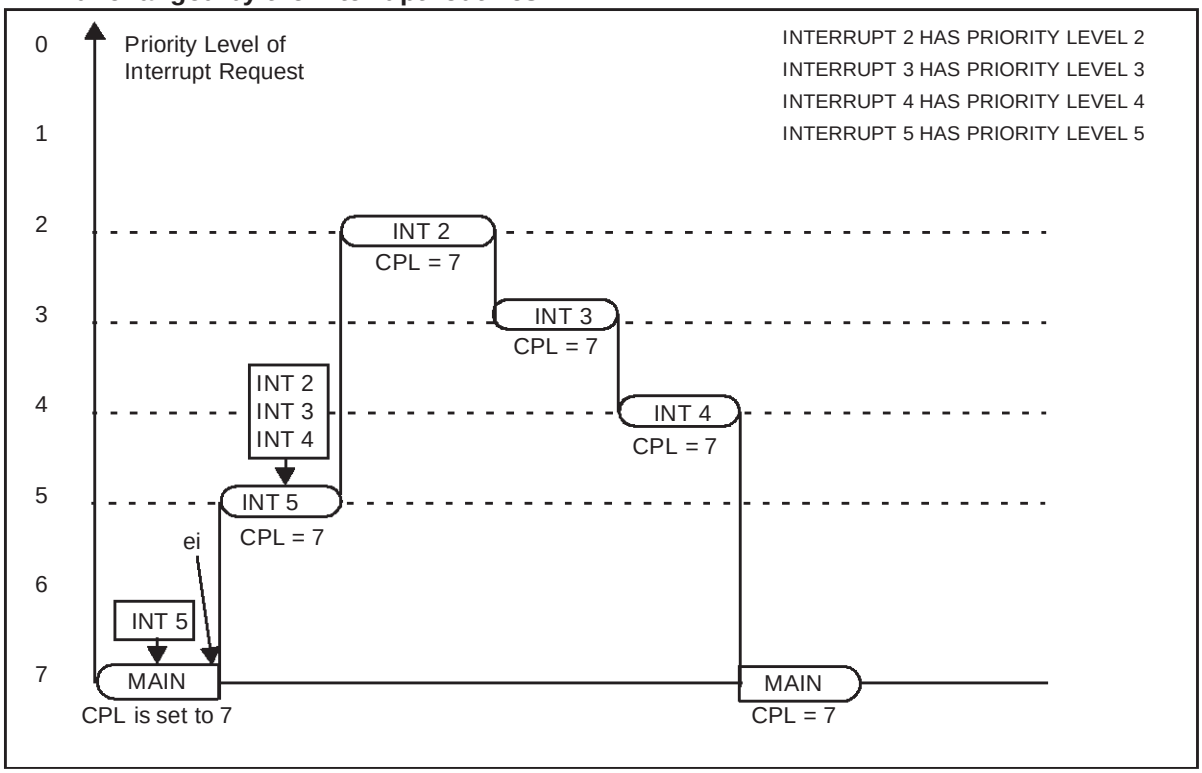
Examples

In the following two examples, three interrupt requests with different priority levels (2, 3 & 4) occur simultaneously during the interrupt 5 service routine.

Example 1

In the first example, (simplest case, Figure 3) the `ei` instruction is not used within the interrupt service routines. This means that no new interrupt can be serviced in the middle of the current one. The interrupt routines will thus be serviced one after another, in the order of their priority, until the main program eventually resumes.

Figure 21. Simple Example of a Sequence of Interrupt Requests with:
- Concurrent mode selected and
- IEN unchanged by the interrupt routines



ARBITRATION MODES (Cont'd)

Example 2

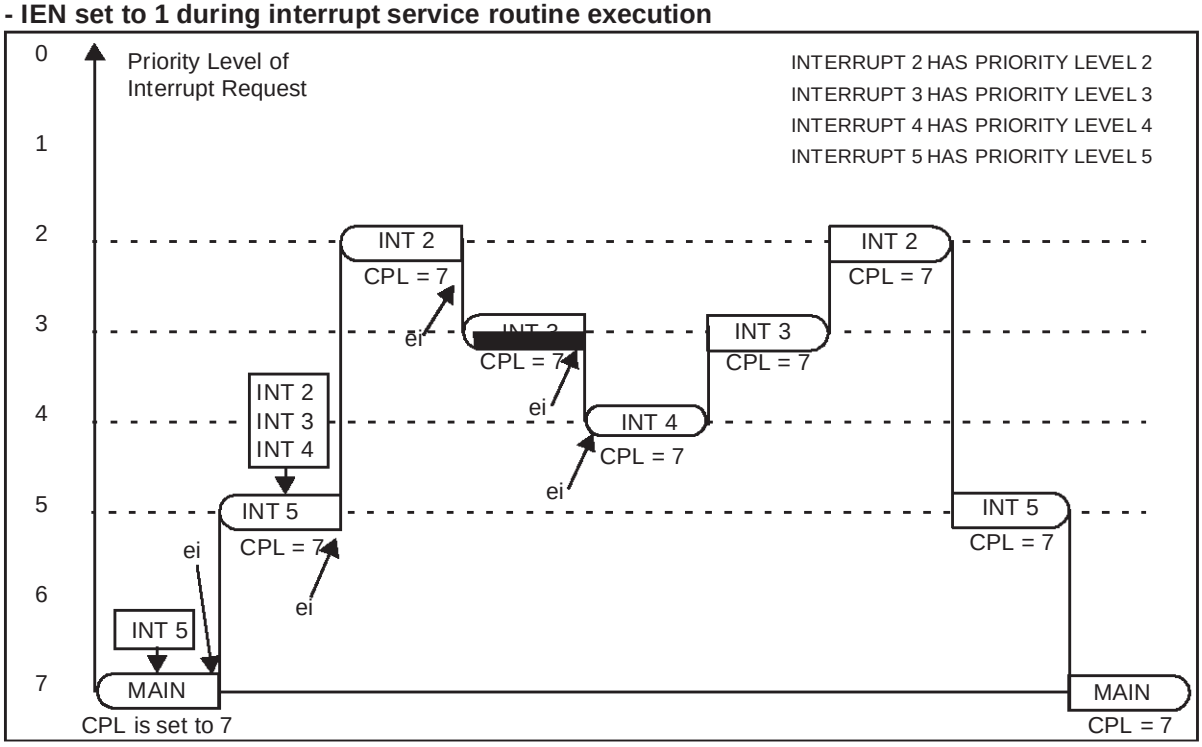
In the second example, (more complex, Figure 4), each interrupt service routine sets Interrupt Enable with the `ei` instruction at the beginning of the routine. Placed here, it minimizes response time for requests with a higher priority than the one being serviced.

The level 2 interrupt routine (with the highest priority) will be acknowledged first, then, when the `ei` instruction is executed, it will be interrupted by the level 3 interrupt routine, which itself will be interrupted by the level 4 interrupt routine. When the level 4 interrupt routine is completed, the level 3 interrupt routine resumes and finally the level 2 interrupt routine. This results in the three interrupt service routines being executed in the opposite order of their priority.

It is therefore recommended to avoid inserting the `ei` instruction in the interrupt service routine in Concurrent mode Use the `ei` instruction only in nested mode.

WARNING: If, in Concurrent Mode, interrupts are nested (by executing `ei` in an interrupt service routine), make sure that either ENCSR is set or CSR=ISR, otherwise the `iret` of the innermost interrupt will make the CPU use CSR instead of ISR before the outermost interrupt service routine is terminated, thus making the outermost routine fail.

Figure 22. Complex Example of a Sequence of Interrupt Requests with:
- Concurrent mode selected
- IEN set to 1 during interrupt service routine execution



ARBITRATION MODES (Cont'd)

4.5.2 Nested Mode

The difference between Nested mode and Concurrent mode, lies in the modification of the Current Priority Level (CPL) during interrupt processing.

The arbitration phase is basically identical to Concurrent mode, however, once the request is acknowledged, the CPL is saved in the Nested Interrupt Control Register (NICR) by setting the NICR bit corresponding to the CPL value (i.e. if the CPL is 3, the bit 3 will be set).

The CPL is then loaded with the priority of the request just acknowledged; the next arbitration cycle is thus performed with reference to the priority of the interrupt service routine currently being executed.

Start of Interrupt Routine

The interrupt cycle performs the following steps:

– All maskable interrupt requests are disabled by clearing CICR.IEN.

– CPL is saved in the special NICR stack to hold the priority level of the suspended routine.

– Priority level of the acknowledged routine is stored in CPL, so that the next request priority will be compared with the one of the routine currently being serviced.

– The PC low byte is pushed onto system stack.

– The PC high byte is pushed onto system stack.

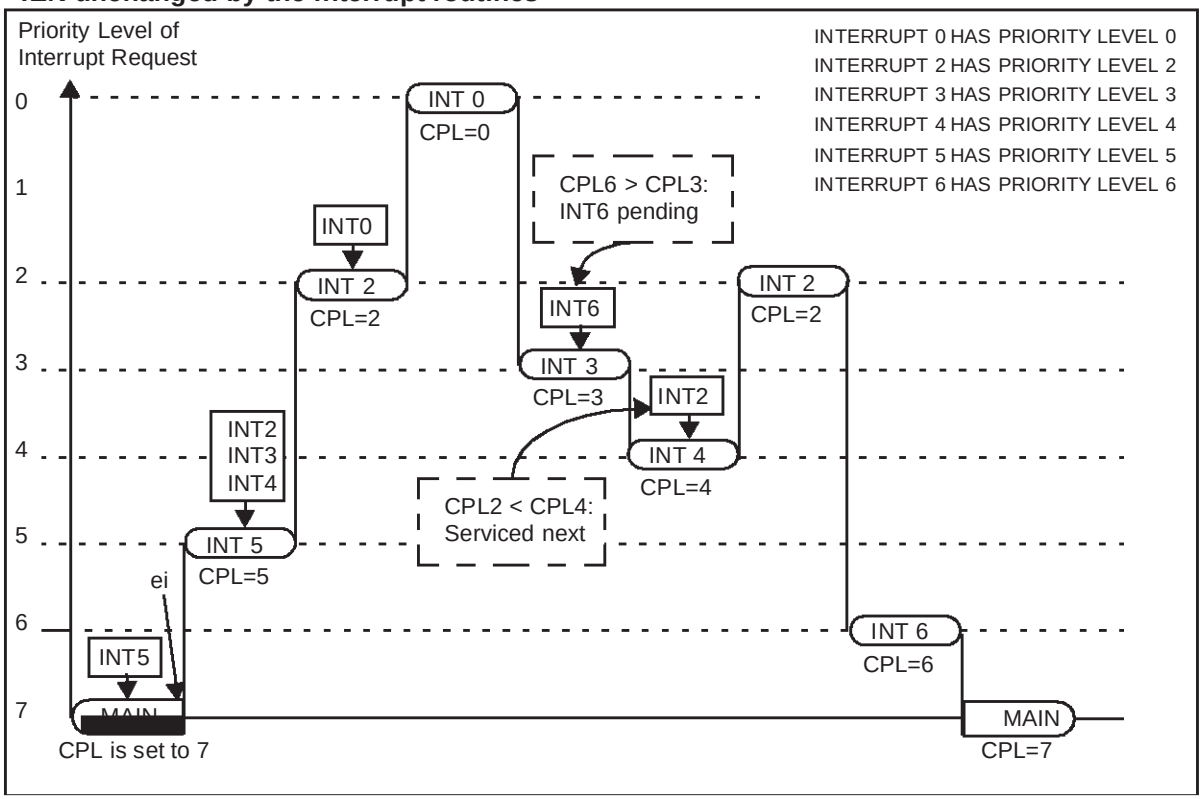
– If ENCSR is set, CSR is pushed onto system stack.

– The Flag register is pushed onto system stack.

– The PC is loaded with the 16-bit vector stored in the Vector Table, pointed to by the IVR.

– If ENCSR is set, CSR is loaded with ISR contents; otherwise ISR is used in place of CSR until `iret` instruction.

Figure 23. Simple Example of a Sequence of Interrupt Requests with:
- Nested mode
- IEN unchanged by the interrupt routines



ARBITRATION MODES (Cont'd)

End of Interrupt Routine

The `iret` Interrupt Return instruction executes the following steps:

– The Flag register is popped from system stack.

– If ENCSR is set, CSR is popped from system stack.

– The PC high byte is popped from system stack.

– The PC low byte is popped from system stack.

– All unmasked Interrupts are enabled by setting the CICR.IEN bit.

– The priority level of the interrupted routine is popped from the special register (NICR) and copied into CPL.

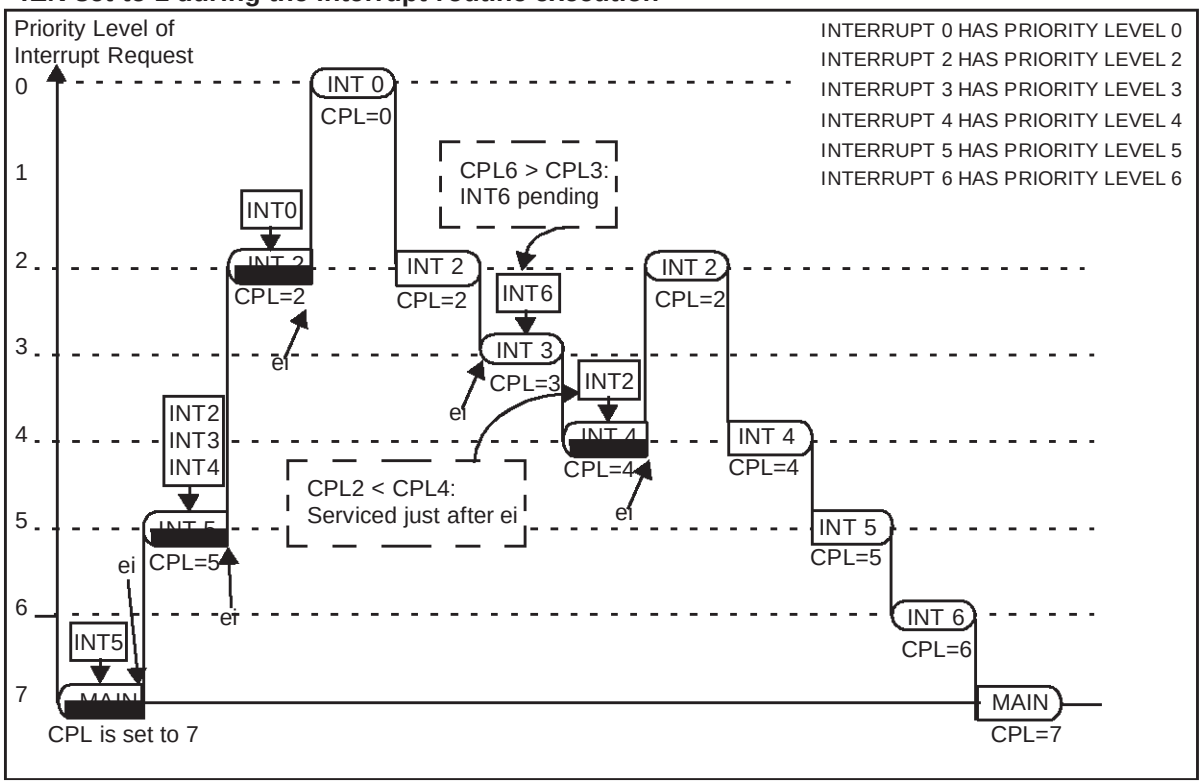
– If ENCSR is reset, CSR is used instead of ISR, unless the program returns to another nested routine.

The suspended routine thus resumes at the interrupted instruction.

Figure 5 contains a simple example, showing that if the `ei` instruction is not used in the interrupt service routines, nested and concurrent modes are equivalent.

Figure 6 contains a more complex example showing how nested mode allows nested interrupt processing (enabled inside the interrupt service routinesi using the `ei` instruction) according to their priority level.

Figure 24. Complex Example of a Sequence of Interrupt Requests with:
- Nested mode
- IEN set to 1 during the interrupt routine execution



4.6 EXTERNAL INTERRUPTS

The standard ST9 core contains 8 external interrupts sources grouped into four pairs.

Table 11. External Interrupt Channel Grouping

External Interrupt	Channel
INT7 INT6	INTD1 INTD0
INT5 INT4	INTC1 INTC0
INT3 INT2	INTB1 INTB0
INT1 INT0	INTA1 INTA0

Each source has a trigger control bit TEA0,..TED1 (R242,EITR.0,..,7 Page 0) to select triggering on the rising or falling edge of the external pin. If the Trigger control bit is set to “1”, the corresponding pending bit IPA0,..,IPD1 (R243,EIPR.0,..,7 Page 0) is set on the input pin rising edge, if it is cleared, the pending bit is set on the falling edge of the input pin. Each source can be individually masked through the corresponding control bit IMA0,..,IMD1 (EIMR.7,..,0). See Figure 8.

The priority level of the external interrupt sources can be programmed among the eight priority levels with the control register EIPLR (R245). The priority level of each pair is software defined using the bits PRL2, PRL1. For each pair, the even channel (A0,B0,C0,D0) of the group has the even priority level and the odd channel (A1,B1,C1,D1) has the odd (lower) priority level.

Figure 25. Priority Level Examples

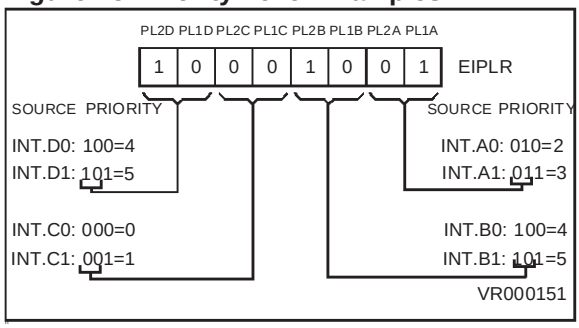


Figure 7 shows an example of priority levels.

Figure 8 gives an overview of the External interrupt control bits and vectors.

– The source of the interrupt channel A0 can be selected between the external pin INT0 (when IA0S = “1”, the reset value) or the On-chip Timer/ Watchdog peripheral (when IA0S = “0”).

– The source of the interrupt channel B0 can be selected between the external pin INT2 (when (SPEN,BMS)=(0,0)) or the on-chip SPI peripheral.

– The source of the interrupt channel C0 can be selected between the external pin INT4 (when INTS = “1”) or the on-chip Standard Timer.

– The source of the interrupt channel D0 can be selected between the external pin INT6 (when INT_SEL = “0”) or the on-chip RCCU.

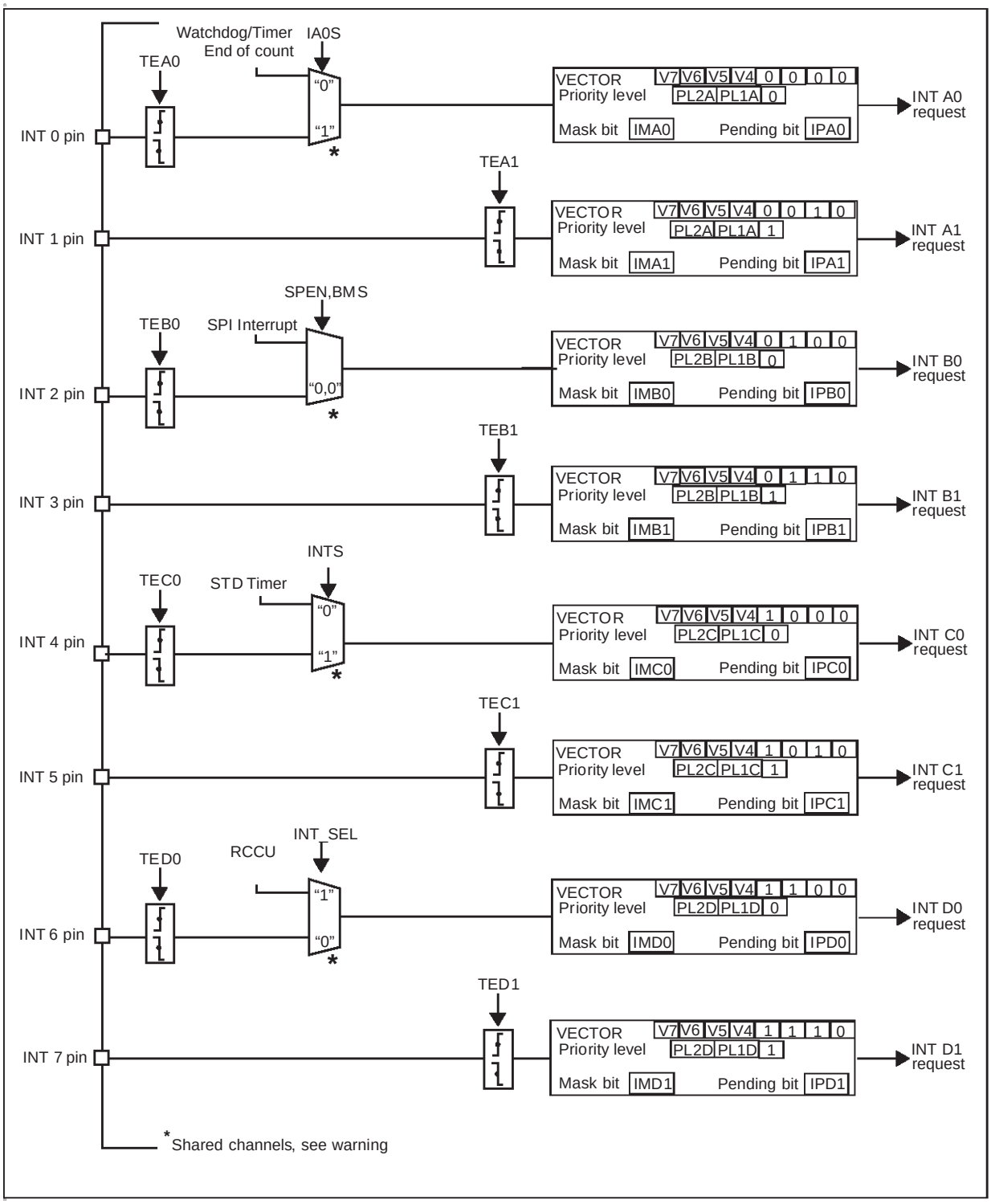
Warning: When using channels shared by both external interrupts and peripherals, special care must be taken to configure their control registers for both peripherals and interrupts.

Table 12. Multiplexed Interrupt Sources

Channel	Internal Interrupt Source	External Interrupt Source
INTA0	Timer/Watchdog	INT0

EXTERNAL INTERRUPTS (Cont'd)

Figure 26. External Interrupts Control Bits and Vectors



4.7 TOP LEVEL INTERRUPT

The Top Level Interrupt channel can be assigned either to the external pin NMI or to the Timer/ Watchdog according to the status of the control bit EIVR.TLIS (R246.2, Page 0). If this bit is high (the reset condition) the source is the external pin NMI. If it is low, the source is the Timer/ Watchdog End Of Count. When the source is the NMI external pin, the control bit EIVR.TLTEV (R246.3; Page 0) selects between the rising (if set) or falling (if reset) edge generating the interrupt request. When the selected event occurs, the CICR.TLIP bit (R230.6) is set. Depending on the mask situation, a Top Level Interrupt request may be generated. Two kinds of masks are available, a Maskable mask and a Non-Maskable mask. The first mask is the CICR.TLI bit (R230.5): it can be set or cleared to enable or disable respectively the Top Level Interrupt request. If it is enabled, the global Enable Interrupt bit, CICR.IEN (R230.4) must also be enabled in order to allow a Top Level Request.

The second mask NICR.TLNM (R247.7) is a set-only mask. Once set, it enables the Top Level Interrupt request independently of the value of CICR.IEN and it cannot be cleared by the program. Only the processor RESET cycle can clear this bit. This does not prevent the user from ignoring some sources due to a change in TLIS.

The Top Level Interrupt Service Routine cannot be interrupted by any other interrupt or DMA request, in any arbitration mode, not even by a subsequent Top Level Interrupt request.

Warning. The interrupt machine cycle of the Top Level Interrupt does not clear the CICR.IEN bit, and the corresponding `iret` does not set it. Furthermore the TLI never modifies the CPL bits and the NICR register.

4.8 ON-CHIP PERIPHERAL INTERRUPTS

The general structure of the peripheral interrupt unit is described here, however each on-chip peripheral has its own specific interrupt unit containing one or more interrupt channels, or DMA channels. Please refer to the specific peripheral chapter for the description of its interrupt features and control registers.

The on-chip peripheral interrupt channels provide the following control bits:

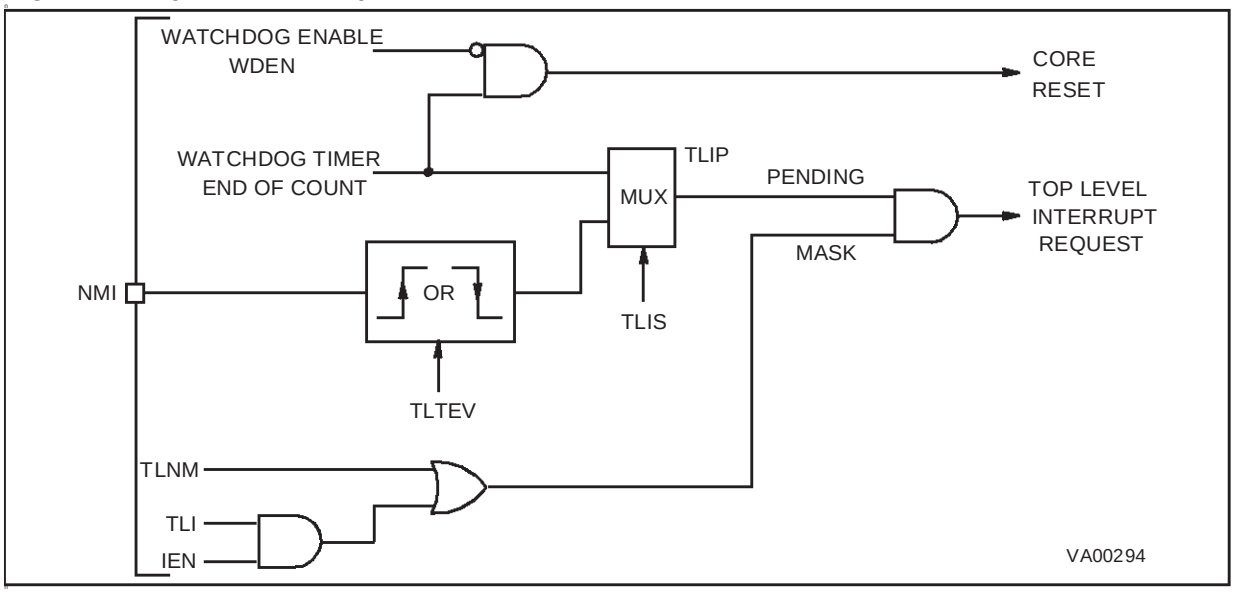
– **Interrupt Pending bit** (IP). Set by hardware when the Trigger Event occurs. Can be set/ cleared by software to generate/cancel pending interrupts and give the status for Interrupt polling.

– **Interrupt Mask bit** (IM). If IM = “0”, no interrupt request is generated. If IM =“1” an interrupt request is generated whenever IP = “1” and CICR.IEN = “1”.

– **Priority Level** (PRL, 3 bits). These bits define the current priority level, PRL=0: the highest priority, PRL=7: the lowest priority (the interrupt cannot be acknowledged)

– **Interrupt Vector Register** (IVR, up to 7 bits). The IVR points to the vector table which itself contains the interrupt routine start address.

Figure 27. Top Level Interrupt Structure



4.9 INTERRUPT RESPONSE TIME

The interrupt arbitration protocol functions completely asynchronously from instruction flow and requires 5 clock cycles. One more CPUCLK cycle is required when an interrupt is acknowledged. Requests are sampled every 5 CPUCLK cycles.

If the interrupt request comes from an external pin, the trigger event must occur a minimum of one INTCLK cycle before the sampling time.

When an arbitration results in an interrupt request being generated, the interrupt logic checks if the current instruction (which could be at any stage of execution) can be safely aborted; if this is the case, instruction execution is terminated immediately and the interrupt request is serviced; if not, the CPU waits until the current instruction is terminated and then services the request. Instruction execution can normally be aborted provided no write operation has been performed.

For an interrupt deriving from an external interrupt channel, the response time between a user event and the start of the interrupt service routine can range from a minimum of 26 clock cycles to a maximum of 55 clock cycles (DIV instruction), 53 clock cycles (DIVWS and MUL instructions) or 49 for other instructions.

For a non-maskable Top Level interrupt, the response time between a user event and the start of the interrupt service routine can range from a minimum of 22 clock cycles to a maximum of 51 clock cycles (DIV instruction), 49 clock cycles (DIVWS and MUL instructions) or 45 for other instructions.

In order to guarantee edge detection, input signals must be kept low/high for a minimum of one INTCLK cycle.

An interrupt machine cycle requires a basic 18 internal clock cycles (CPUCLK), to which must be added a further 2 clock cycles if the stack is in the Register File. 2 more clock cycles must further be added if the CSR is pushed (ENCSR =1).

The interrupt machine cycle duration forms part of the two examples of interrupt response time previously quoted; it includes the time required to push values on the stack, as well as interrupt vector handling.

In Wait for Interrupt mode, a further cycle is required as wake-up delay.

4.10 INTERRUPT REGISTERS

CENTRAL INTERRUPT CONTROL REGISTER (CICR)
R230 - Read/Write
Register Group: System
Reset value: 1000 0111 (87h)

7							0
GCEN	TLIP	TLI	IEN	IAM	CPL2	CPL1	CPL0

Bit 7 = **GCEN**: *Global Counter Enable.*
This bit enables the 16-bit Multifunction Timer peripheral.
0: MFT disabled
1: MFT enabled

Bit 6 = **TLIP**: *Top Level Interrupt Pending.*
This bit is set by hardware when Top Level Interrupt (TLI) trigger event occurs. It is cleared by hardware when a TLI is acknowledged. It can also be set by software to implement a software TLI.
0: No TLI pending
1: TLI pending

Bit 5 = **TLI**: *Top Level Interrupt.*
This bit is set and cleared by software.
0: A Top Level Interrupt is generared when TLIP is set, only if TLNM=1 in the NICR register (independently of the value of the IEN bit).
1: A Top Level Interrupt request is generated when IEN=1 and the TLIP bit are set.

Bit 4 = **IEN**: *Interrupt Enable.*
This bit is cleared by the interrupt machine cycle (except for a TLI).
It is set by the `iret` instruction (except for a return from TLI).
It is set by the `EI` instruction.
It is cleared by the `DI` instruction.
0: Maskable interrupts disabled
1: Maskable Interrupts enabled

Note: The IEN bit can also be changed by software using any instruction that operates on register CICR, however in this case, take care to avoid spurious interrupts, since IEN cannot be cleared in the middle of an interrupt arbitration. Only modify the IEN bit when interrupts are disabled or when no peripheral can generate interrupts. For example, if the state of IEN is not known in advance, and its value must be restored from a previous push of CICR on the stack, use the sequence`DI; POP CICR` to make sure that no interrupts are being arbitrated when CICR is modified.

Bit 3 = **IAM**: *Interrupt Arbitration Mode.*
This bit is set and cleared by software.
0: Concurrent Mode
1: Nested Mode

Bit 2:0 = **CPL[2:0]:** *Current Priority Level.*
These bits define the Current Priority Level. CPL=0 is the highest priority. CPL=7 is the lowest priority. These bits may be modified directly by the interrupt hardware when Nested Interrupt Mode is used.

EXTERNAL INTERRUPT TRIGGER REGISTER (EITR)
R242 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
TED1	TED0	TEC1	TEC0	TEB1	TEB0	TEA1	TEA0

Bit 7 = **TED1**: *INTD1 Trigger Event*
Bit 6 = **TED0**: *INTD0 Trigger Event*
Bit 5 = **TEC1**: *INTC1 Trigger Event*
Bit 4 = **TEC0**: *INTC0 Trigger Event*
Bit 3 = **TEB1**: *INTB1 Trigger Event*
Bit 2 = **TEB0**: *INTB0 Trigger Event*
Bit 1 = **TEA1**: *INTA1 Trigger Event*
Bit 0 = **TEA0**: *INTA0 Trigger Event*

These bits are set and cleared by software.
0: Select falling edge as interrupt trigger event
1: Select rising edge as interrupt trigger event

INTERRUPT REGISTERS (Cont'd)

EXTERNAL INTERRUPT PENDING REGISTER (EIPR)
R243 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
IPD1	IPD0	IPC1	IPC0	IPB1	IPB0	IPA1	IPA0

Bit 7 = **IPD1**: *INTD1 Interrupt Pending bit*
Bit 6 = **IPD0**: *INTD0 Interrupt Pending bit*
Bit 5 = **IPC1**: *INTC1 Interrupt Pending bit*
Bit 4 = **IPC0**: *INTC0 Interrupt Pending bit*
Bit 3 = **IPB1**: *INTB1 Interrupt Pending bit*
Bit 2 = **IPB0**: *INTB0 Interrupt Pending bit*
Bit 1 = **IPA1**: *INTA1 Interrupt Pending bit*
Bit 0 = **IPA0**: *INTA0 Interrupt Pending bit*

These bits are set by hardware on occurrence of a trigger event (as specified in the EITR register) and are cleared by hardware on interrupt acknowledge. They can also be set by software to implement a software interrupt.
0: No interrupt pending
1: Interrupt pending

EXTERNAL INTERRUPT MASK-BIT REGISTER (EIMR)
R244 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
IMD1	IMD0	IMC1	IMC0	IMB1	IMB0	IMA1	IMA0

Bit 7 = **IMD1**: *INTD1 Interrupt Mask*
Bit 6 = **IMD0**: *INTD0 Interrupt Mask*
Bit 5 = **IMC1**: *INTC1 Interrupt Mask*
Bit 4 = **IMC0**: *INTC0 Interrupt Mask*
Bit 3 = **IMB1**: *INTB1 Interrupt Mask*
Bit 2 = **IMB0**: *INTB0 Interrupt Mask*
Bit 1 = **IMA1**: *INTA1 Interrupt Mask*
Bit 0 = **IMA0**: *INTA0 Interrupt Mask*

These bits are set and cleared by software.
0: Interrupt masked
1: Interrupt not masked (an interrupt is generated if the IPxx and IEN bits = 1)

EXTERNAL INTERRUPT PRIORITY LEVEL REGISTER (EIPLR)
R245 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh)

7							0
PL2D	PL1D	PL2C	PL1C	PL2B	PL1B	PL2A	PL1A

Bit 7:6 = **PL2D, PL1D:** *INTD0, D1 Priority Level.*
Bit 5:4 = **PL2C, PL1C**: *INTC0, C1 Priority Level.*
Bit 3:2 = **PL2B, PL1B**: *INTB0, B1 Priority Level.*
Bit 1:0 = **PL2A, PL1A**: *INTA0, A1 Priority Level.*

These bits are set and cleared by software.

The priority is a three-bit value. The LSB is fixed by hardware at 0 for Channels A0, B0, C0 and D0 and at 1 for Channels A1, B1, C1 and D1.

PL2x	PL1x	Hardware bit	Priority
0	0	0 1	0 (Highest) 1
0	1	0 1	2 3
1	0	0 1	4 5
1	1	0 1	6 7 (Lowest)

INTERRUPT REGISTERS (Cont'd)

EXTERNAL INTERRUPT VECTOR REGISTER (EIVR)

R246 - Read/Write
Register Page: 0
Reset value: xxxx 0110b (x6h)

7							0
V7	V6	V5	V4	TLTEV	TLIS	IAOS	EWEN

Bit 7:4 = **V[7:4]**: *Most significant nibble of External Interrupt Vector.*
These bits are not initialized by reset. For a representation of how the full vector is generated from V[7:4] and the selected external interrupt channel, refer to Figure 8.

Bit 3 = **TLTEV**: *Top Level Trigger Event bit.*
This bit is set and cleared by software.
0: Select falling edge as NMI trigger event
1: Select rising edge as NMI trigger event

Bit 2 = **TLIS**: *Top Level Input Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is TL interrupt source
1: NMI is TL interrupt source

Bit 1 = **IA0S**: *Interrupt Channel A0 Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is INTA0 source
1: External Interrupt pin is INTA0 source

Bit 0 = **EWEN**: *External Wait Enable.*
This bit is set and cleared by software.
0: WAITN pin disabled
1: WAITN pin enabled (to stretch the external memory access cycle).

Note: For more details on Wait mode refer to the section describing the WAITN pin in the External Memory Chapter.

NESTED INTERRUPT CONTROL (NICR)

R247 - Read/Write
Register Page: 0
Reset value: 0000 0000 (00h)

7							0
TLNM	HL6	HL5	HL4	HL3	HL2	HL1	HL0

Bit 7 = **TLNM**: *Top Level Not Maskable*
This bit is set by software and cleared only by a hardware reset.
0: Top Level Interrupt Maskable. A top level request is generated if the IEN, TLI and TLIP bits =1
1: Top Level Interrupt Not Maskable. A top level request is generated if the TLIP bit =1

Bit 6:0 = **HL[6:0]**: *Hold Level* x
These bits are set by hardware when, in Nested Mode, an interrupt service routine at level x is interrupted from a request with higher priority (other than the Top Level interrupt request). They are cleared by hardware at the `iret` execution when the routine at level x is recovered.

INTERRUPT REGISTERS (Cont'd)

EXTERNAL MEMORY REGISTER 2(EMR2)

R246 - Read/Write
Register Page: 21
Reset value: 0000 1111 (0Fh)

7							0
0	ENCSR	0	0	1	1	1	1

Bit 7, 5:0 = Reserved, keep in reset state. Refer to the external Memory Interface Chapter.

Bit 6 = **ENCSR**: *Enable Code Segment Register.* This bit is set and cleared by software. It affects the ST9 CPU behaviour whenever an interrupt request is issued.

0: The CPU works in original ST9 compatibility mode. For the duration of the interrupt service routine, ISR is used instead of CSR, and the interrupt stack frame is identical to that of the original ST9: only the PC and Flags are pushed. This avoids saving the CSR on the stack in the event of an interrupt, thus ensuring a faster interrupt response time. The drawback is that it is not possible for an interrupt service routine to perform inter-segment calls or jumps: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code segment size for all interrupt service routines is thus limited to 64K bytes.

1: ISR is only used to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and flags, and CSR is then loaded with the contents of ISR. In this case, `iret` will also restore CSR from the stack. This approach allows interrupt service routines to access the entire 4 Mbytes of address space; the drawback is that the interrupt response time is slightly increased, because of the need to also save CSR on the stack. Full compatibility with the original ST9 is lost in this case, because the interrupt stack frame is different; this difference, however, should not affect the vast majority of programs.

5 ON-CHIP DIRECT MEMORY ACCESS (DMA)

5.1 INTRODUCTION

The ST9 includes on-chip Direct Memory Access (DMA) in order to provide high-speed data transfer between peripherals and memory or Register File. Multi-channel DMA is fully supported by peripherals having their own controller and DMA channel(s). Each DMA channel transfers data to or from contiguous locations in the Register File, or in Memory. The maximum number of bytes that can be transferred per transaction by each DMA channel is 222 with the Register File, or 65536 with Memory.

The DMA controller in the Peripheral uses an indirect addressing mechanism to DMA Pointers and Counter Registers stored in the Register File. This is the reason why the maximum number of transactions for the Register File is 222, since two Registers are allocated for the Pointer and Counter. Register pairs are used for memory pointers and counters in order to offer the full 65536 byte and count capability.

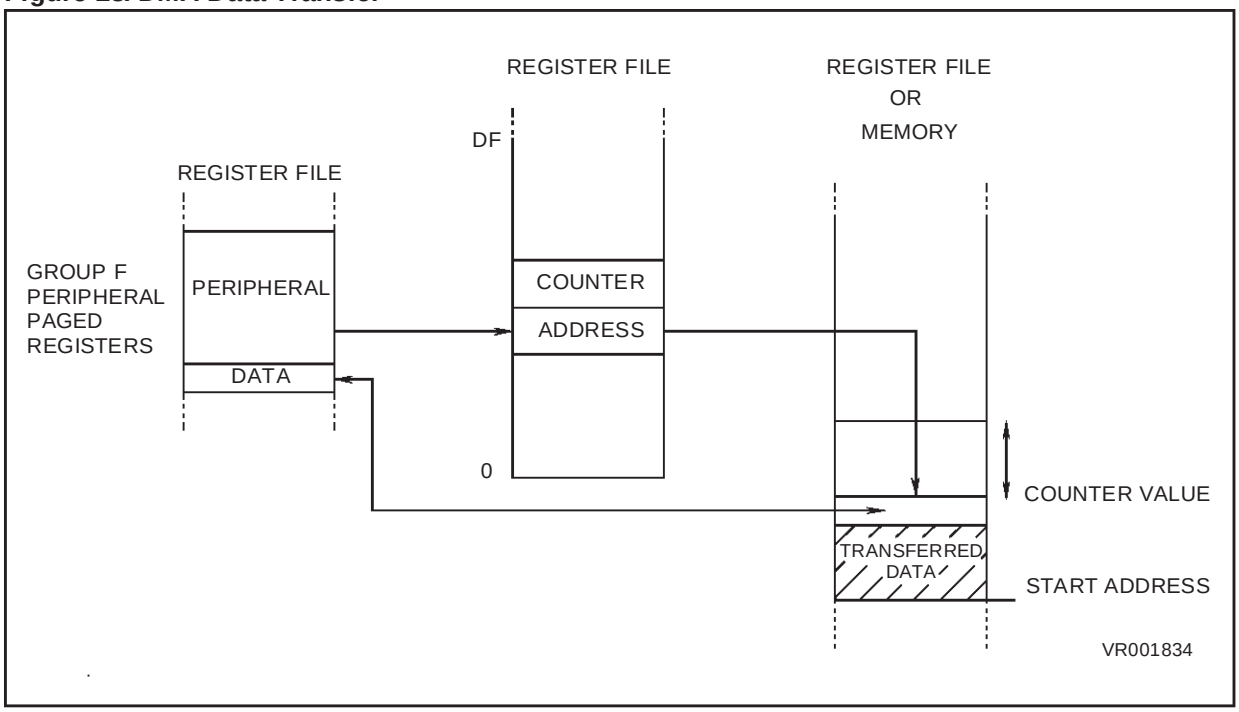
5.2 DMA PRIORITY LEVELS

The 8 priority levels used for interrupts are also used to prioritize the DMA requests, which are arbitrated in the same arbitration phase as interrupt requests. If the event occurrence requires a DMA transaction, this will take place at the end of the current instruction execution. When an interrupt and a DMA request occur simultaneously, on the same priority level, the DMA request is serviced before the interrupt.

An interrupt priority request must be strictly higher than the CPL value in order to be acknowledged, whereas, for a DMA transaction request, it must be equal to or higher than the CPL value in order to be executed. Thus only DMA transaction requests can be acknowledged when the CPL=0.

DMA requests do not modify the CPL value, since the DMA transaction is not interruptable.

Figure 28. DMA Data Transfer



5.3 DMA TRANSACTIONS

The purpose of an on-chip DMA channel is to transfer a block of data between a peripheral and the Register File, or Memory. Each DMA transfer consists of three operations:

– A load from/to the peripheral data register to/ from a location of Register File (or Memory) addressed through the DMA Address Register (or Register pair)

– A post-increment of the DMA Address Register (or Register pair)

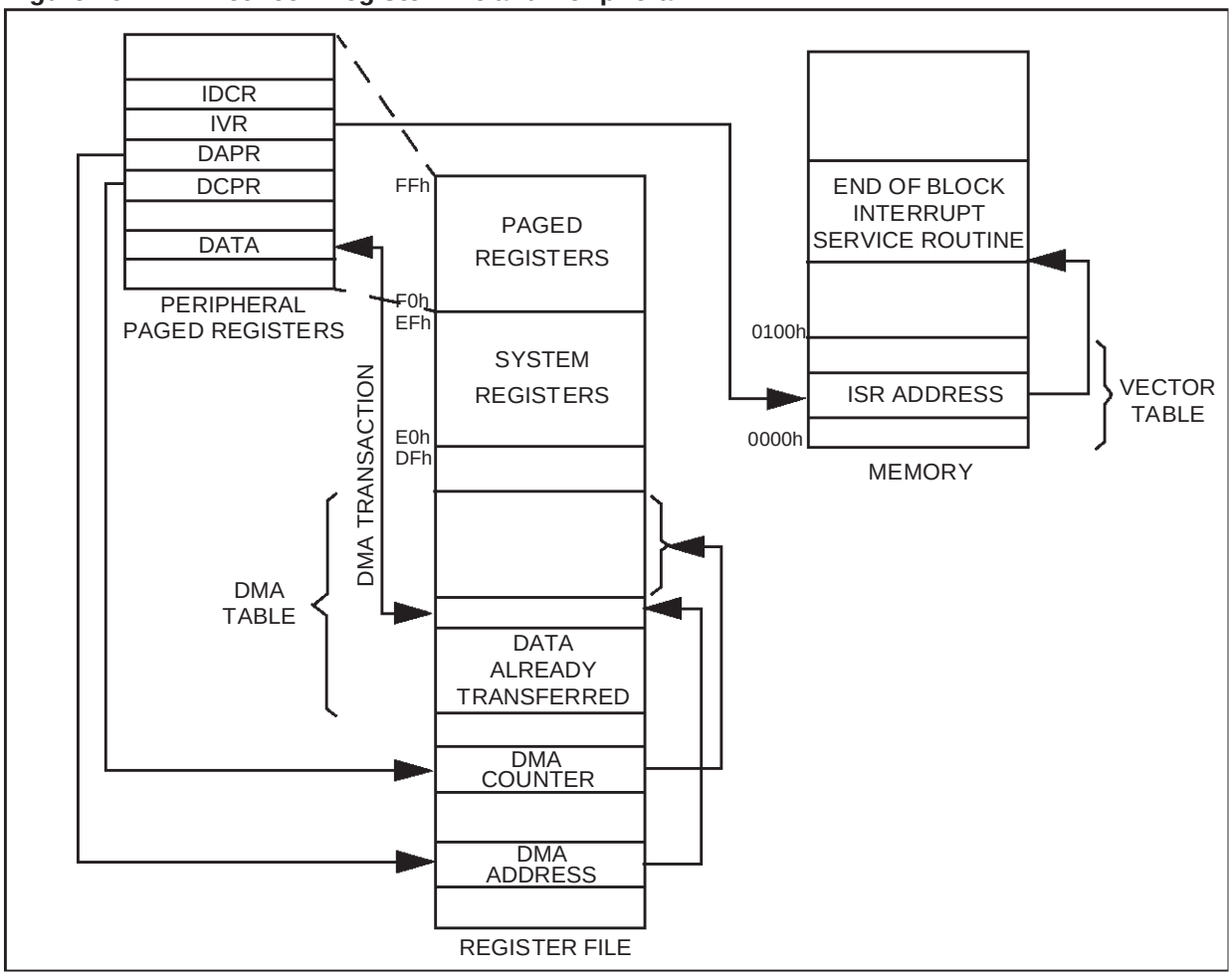
– A post-decrement of the DMA transaction counter, which contains the number of transactions that have still to be performed.

If the DMA transaction is carried out between**the peripheral and the Register File**(Figure 29), one register is required to hold the DMA Address, and one to hold the DMA transaction counter. These two registers must be located in the Register File: the DMA Address Register in the even address register, and the DMA Transaction Counter in the next register (odd address). They are pointed to by the DMA Transaction Counter Pointer Register (DCPR), located in the peripheral's paged registers. In order to select a DMA transaction with the Register File, the control bit DCPR.RM (bit 0 of DCPR) must be set.

If the transaction is made between**the peripheral and Memory**, a register pair (16 bits) is required for the DMA Address and the DMA Transaction Counter (Figure 30). Thus, two register pairs must be located in the Register File.

The DMA Transaction Counter is pointed to by the DMA Transaction Counter Pointer Register (DCPR), the DMA Address is pointed to by the DMA Address Pointer Register (DAPR),both DCPR and DAPR are located in the paged registers of the peripheral.

Figure 29. DMA Between Register File and Peripheral



DMA TRANSACTIONS (Cont'd)

When selecting the DMA transaction with memory, bit DCPR.RM (bit 0 of DCPR) must be cleared.

To select between using the ISR or the DMASR register to extend the address, (see Memory Management Unit chapter), the control bit DAPR.PS (bit 0 of DAPR) must be cleared or set respectively.

The DMA transaction Counter must be initialized with the number of transactions to perform and will be decremented after each transaction. The DMA Address must be initialized with the starting address of the DMA table and is increased after each transaction. These two registers must be located between addresses 00h and DFh of the Register File.

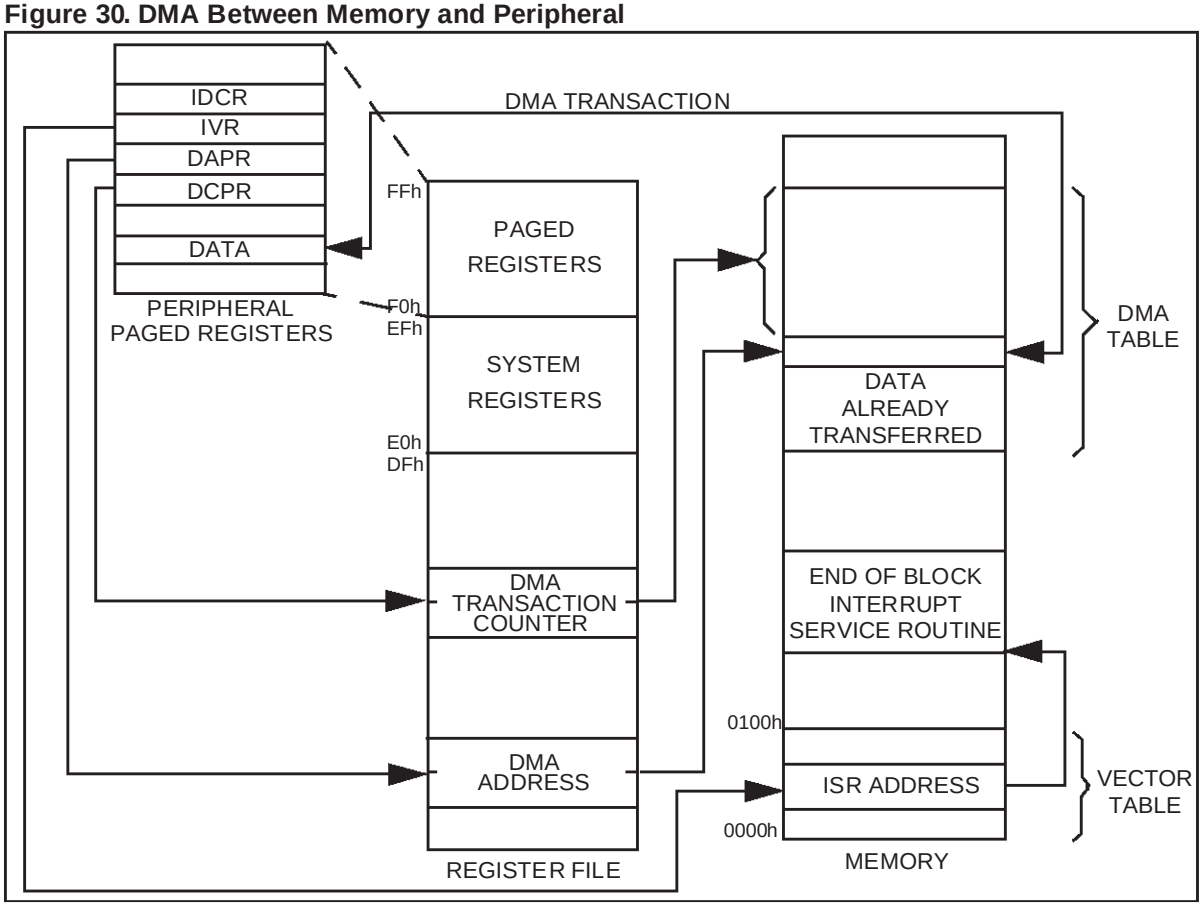
Once a DMA channel is initialized, a transfer can start. The direction of the transfer is automatically defined by the type of peripheral and programming mode.

Once the DMA table is completed (the transaction counter reaches 0 value), an Interrupt request to the CPU is generated.

When the Interrupt Pending (IP) bit is set by a hardware event (or by software), and the DMA Mask bit (DM) is set, a DMA request is generated. If the Priority Level of the DMA source is higher than, or equal to, the Current Priority Level (CPL), the DMA transfer is executed at the end of the current instruction. DMA transfers read/write data from/to the location pointed to by the DMA Address Register, the DMA Address register is incremented and the Transaction Counter Register is decremented. When the contents of the Transaction Counter are decremented to zero, the DMA Mask bit (DM) is cleared and an interrupt request is generated, according to the Interrupt Mask bit (End of Block interrupt). This End-of-Block interrupt request is taken into account, depending on the PRL value.

***WARNING**. DMA requests are not acknowledged if the top level interrupt service is in progress.*

Figure 30. DMA Between Memory and Peripheral



DMA TRANSACTIONS (Cont'd)

5.4 DMA CYCLE TIME

The interrupt and DMA arbitration protocol functions completely asynchronously from instruction flow.

Requests are sampled every 5 CPUCLK cycles.

DMA transactions are executed if their priority allows it.

A DMA transfer with the Register file requires 8 CPUCLK cycles.

A DMA transfer with memory requires 16 CPUCLK cycles, plus any required wait states.

5.5 SWAP MODE

An extra feature which may be found on the DMA channels of some peripherals (e.g. the MultiFunction Timer) is the Swap mode. This feature allows transfer from two DMA tables alternatively. All the DMA descriptors in the Register File are thus doubled. Two DMA transaction counters and two DMA address pointers allow the definition of two fully independent tables (they only have to belong to the same space, Register File or Memory). The DMA transaction is programmed to start on one of the two tables (say table 0) and, at the end of the block, the DMA controller automatically swaps to the other table (table 1) by pointing to the other DMA descriptors. In this case, the DMA mask (DM bit) control bit is not cleared, but the End Of Block interrupt request is generated to allow the optional updating of the first data table (table 0).

Until the swap mode is disabled, the DMA controller will continue to swap between DMA Table 0 and DMA Table 1.

5.6 DMA REGISTERS

As each peripheral DMA channel has its own specific control registers, the following register list should be considered as a general example. The names and register bit allocations shown here may be different from those found in the peripheral chapters.

DMA COUNTER POINTER REGISTER(DCPR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
C7	C6	C5	C4	C3	C2	C1	RM

Bit 7:1 = **C[7:1]**: *DMA Transaction Counter Pointer.*

Software should write the pointer to the DMA Transaction Counter in these bits.

Bit 0 = **RM**: *Register File/Memory Selector.*
This bit is set and cleared by software.
0: DMA transactions are with memory (see also DAPR.DP)
1: DMA transactions are with the Register File

GENERIC EXTERNAL PERIPHERAL INTERRUPT AND DMA CONTROL(IDCR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
		IP	DM	IM	PRL2	PRL1	PRL0

Bit 5 = **IP**: *Interrupt Pending.*

This bit is set by hardware when the Trigger Event occurs. It is cleared by hardware when the request is acknowledged. It can be set/cleared by software in order to generate/cancel a pending request.
0: No interrupt pending
1: Interrupt pending

Bit 4 = **DM**: *DMA Request Mask.*

This bit is set and cleared by software. It is also cleared when the transaction counter reaches zero (unless SWAP mode is active).
0: No DMA request is generated when IP is set.
1: DMA request is generated when IP is set

Bit 3 = **IM**: *End of block Interrupt Mask.*
This bit is set and cleared by software.
0: No End of block interrupt request is generated when IP is set
1: End of Block interrupt is generated when IP is set. DMA requests depend on the DM bit value as shown in the table below.

DM	IM	Meaning
1	0	A DMA request generated without End of Block interrupt when IP=1
1	1	A DMA request generated with End of Block interrupt when IP=1
0	0	No End of block interrupt or DMA request is generated when IP=1
0	1	An End of block Interrupt is generated without associated DMA request (not used)

Bit 2:0 = **PRL[2:0]**: *Source Priority Level.*
These bits are set and cleared by software. Refer to Section 5.2 DMA PRIORITY LEVELS for a description of priority levels.

PRL2	PRL1	PRL0	Source Priority Level
0	0	0	0 Highest
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7 Lowest

DMA ADDRESS POINTER REGISTER(DAPR)
Read/Write
Address set by Peripheral
Reset value: undefined

7							0
A7	A6	A5	A4	A3	A2	A1	DP

Bit 7:1 = **A[7:1]**: *DMA Address Register(s) Pointer*

Software should write the pointer to the DMA Address Register(s) in these bits.

Bit 0 = **PS**: *Memory Segment Pointer Selector.*
This bit is set and cleared by software. It is only meaningful if DAPR.RM=0.
0: The ISR register is used to extend the address of data transferred by DMA (see MMU chapter).
1: The DMASR register is used to extend the address of data transferred by DMA (see MMU chapter).

6 RESET AND CLOCK CONTROL UNIT (RCCU)

6.1 INTRODUCTION

The Reset and Clock Control Unit (RCCU) comprises two distinct sections:

– the Clock Control Unit, which generates and manages the internal clock signals.

– the Reset/Stop Manager, which detects and flags Hardware, Software and Watchdog generated resets.

On ST9 devices where the external Stop pin is available, this circuit also detects and manages the externally triggered Stop mode, during which all oscillators are frozen in order to achieve the lowest possible power consumption.

6.2 CLOCK CONTROL UNIT

The Clock Control Unit generates the internal clocks for the CPU core (CPUCLK) and for the on-chip peripherals (INTCLK). The Clock Control Unit may be driven by an external crystal circuit, connected to the OSCIN and OSCOUT pins, or by an external pulse generator, connected to OSCIN (see Figure 37 and Figure 39).

6.2.1 Clock Control Unit Overview

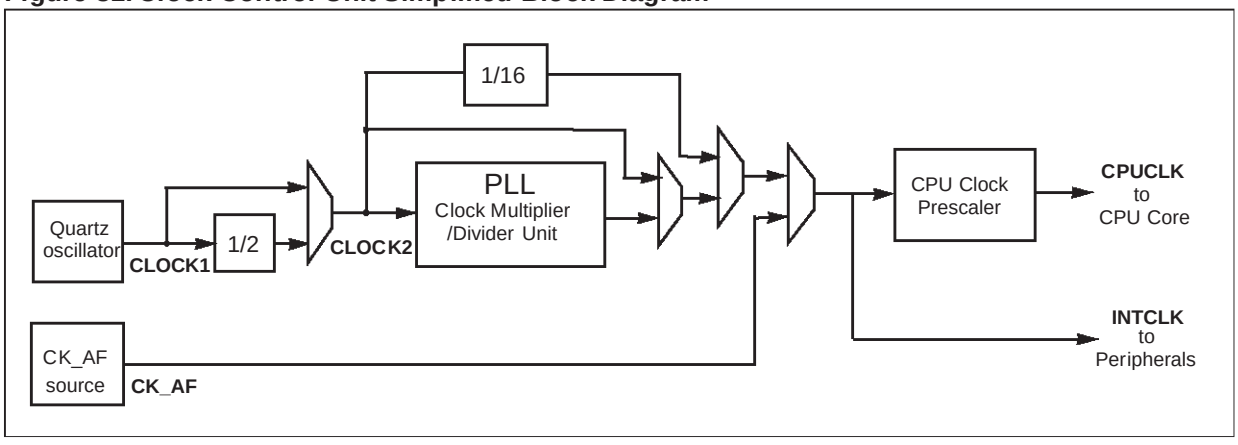
As shown in Figure 31, a programmable divider can divide the CLOCK1 input clock signal by two. The divide-by-two is recommended in order to ensure a 50% duty cycle signal driving the PLL multiplier circuit. The resulting signal, CLOCK2, is the reference input clock to the programmable Phase Locked Loop frequency multiplier, which is capable of multiplying the clock frequency by a factor of 6, 8, 10 or 14; the multiplied clock is then divided by a programmable divider, by a factor of 1 to 7. By this means, the ST9 can operate with cheaper, medium frequency (3-5 MHz) crystals, while still providing a high frequency internal clock for maximum system performance; the range of available multiplication and division factors allow a great number of operating clock frequencies to be derived from a single crystal frequency.

For low power operation, especially in Wait for Interrupt mode, the Clock Multiplier unit may be turned off, whereupon the output clock signal may be programmed as CLOCK2 divided by 16. For further power reduction, a low frequency external clock connected to the CK_AF pin may be selected, whereupon the crystal controlled main oscillator may be turned off.

The internal system clock, INTCLK, is routed to all on-chip peripherals, as well as to the programmable Clock Prescaler Unit which generates the clock for the CPU core (CPUCLK).

The Clock Prescaler is programmable and can slow the CPU clock by a factor of up to 8, allowing the programmer to reduce CPU processing speed, and thus power consumption, while maintaining a high speed clock to the peripherals. This is particularly useful when little actual processing is being done by the CPU and the peripherals are doing most of the work.

Figure 31. Clock Control Unit Simplified Block Diagram



6.3 CLOCK MANAGEMENT

The various programmable features and operating modes of the CCU are handled by four registers:

– **MODER** (Mode Register)
This is a System Register (R235, Group E).

The input clock divide-by-two and the CPU clock prescaler factors are handled by this register.

– **CLKCTL** (Clock Control Register)
This is a Paged Register (R240, Page 55).

The low power modes and the interpretation of the HALT instruction are handled by this register.

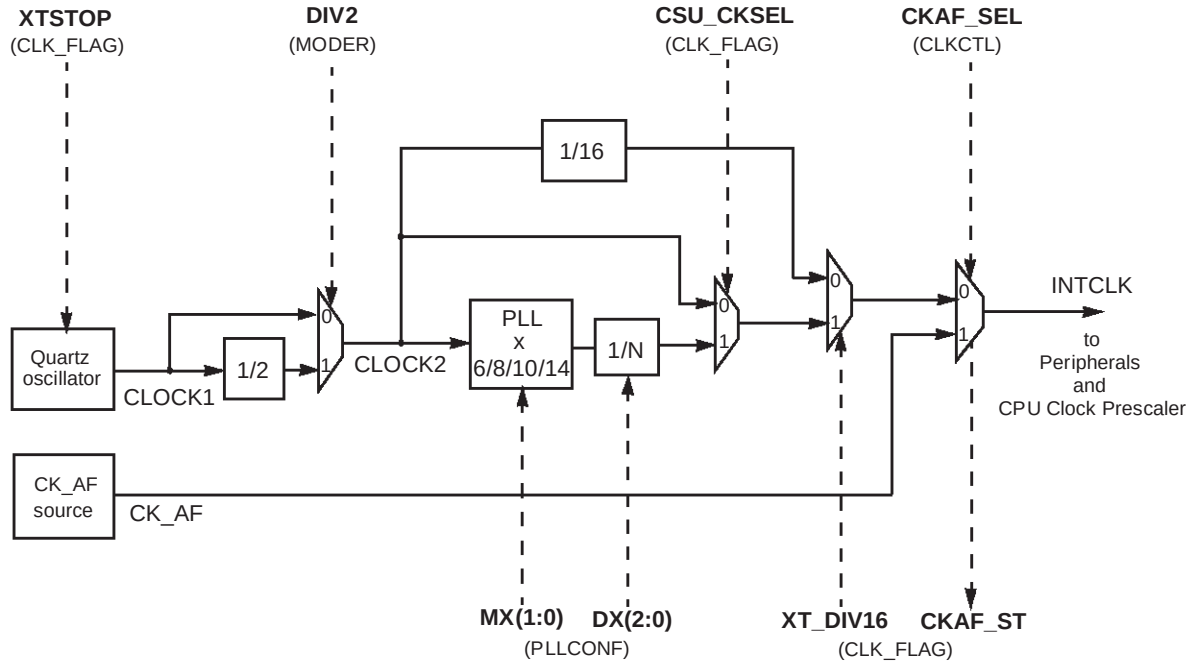
– **CLK_FLAG** (Clock Flag Register)
This is a Paged Register (R242, Page 55).

This register contains various status flags, as well as control bits for clock selection.

– **PLLCONF** (PLL Configuration Register)
This is a Paged Register (R246, Page 55).

The PLL multiplication and division factors are programmed in this register.

Figure 32. Clock Control Unit Programming



Wait for Interrupt and Low Power Modes:

LPOWFI (CLKCTL) selects Low Power operation automatically on entering WFI mode.
WFI_CKSEL (CLKCTL) selects the CK_AF clock automatically, if present, on entering WFI mode.
XTSTOP (CLK_FLAG) automatically stops the Xtal oscillator when the CK_AF clock is present and selected.

CLOCK MANAGEMENT (Cont'd)

6.3.1 PLL Clock Multiplier Programming

The CLOCK1 signal generated by the oscillator drives a programmable divide-by-two circuit. If the DIV2 control bit in MODER is set (Reset Condition), CLOCK2, is equal to CLOCK1 divided by two; if DIV2 is reset, CLOCK2 is identical to CLOCK1. Since the input clock to the Clock Multiplier circuit requires a 50% duty cycle for correct PLL operation, the divide by two circuit should be enabled when a crystal oscillator is used, or when the external clock generator does not provide a 50% duty cycle. In practice, the divide-by-two is virtually always used in order to ensure a 50% duty cycle signal to the PLL multiplier circuit.

When the PLL is active, it multiplies CLOCK2 by 6, 8, 10 or 14, depending on the status of the MX0 -1 bits in PLLCONF. The multiplied clock is then divided by a factor in the range 1 to 7, determined by the status of the DX0-2 bits; when these bits are programmed to 111, the PLL is switched off.

Following a RESET phase, programming bits DX0-2 to a value different from 111 will turn the PLL on. After allowing a stabilisation period for the PLL, setting the CSU_CKSEL bit in the CLK_FLAG Register selects the multiplier clock.

The maximum frequency allowed for INTCLK is 24 MHz for 5V operation, and 16 MHz for 3V operation. Care is required, when programming the PLL multiplier and divider factors, not to exceed the maximum permissible operating frequency for INTCLK, according to supply voltage.

The ST9 being a static machine, there is no lower limit for INTCLK. However, below 1MHz, A/D converter precision (if present) decreases.

6.3.2 CPU Clock Prescaling

The system clock, INTCLK, which may be the output of the PLL clock multiplier, CLOCK2, CLOCK2/16 or CK_AF, drives a programmable prescaler which generates the basic time base, CPUCLK, for the instruction executer of the ST9 CPU core. This allows the user to slow down program execution during non processor intensive routines, thus reducing power dissipation.

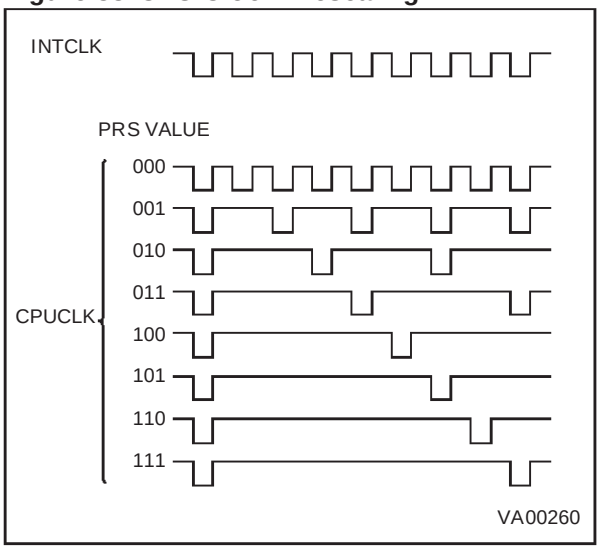
The internal peripherals are not affected by the CPUCLK prescaler and continue to operate at the full INTCLK frequency. This is particularly useful when little processing is being done and the peripherals are doing most of the work.

The prescaler divides the input clock by the value programmed in the control bits PRS2,1,0 in the MODER register. If the prescaler value is zero, no prescaling takes place, thus CPUCLK has the same period and phase as INTCLK. If the value is different from 0, the prescaling is equal to the value plus one, ranging thus from two (PRS2,1,0 = 1) to eight (PRS2,1,0 = 7).

The clock generated is shown in Figure 33, and it will be noted that the prescaling of the clock does not preserve the 50% duty cycle, since the high level is stretched to replace the missing cycles.

This is analogous to the introduction of wait cycles for access to external memory. When External Memory Wait or Bus Request events occur, CPUCLK is stretched at the high level for the whole period required by the function.

Figure 33. CPU Clock Prescaling



6.3.3 Peripheral Clock

The system clock, INTCLK, which may be the output of the PLL clock multiplier, CLOCK2, CLOCK2/16 or CK_AF, is also routed to all ST9 on-chip peripherals and acts as the central timebase for all timing functions.

CLOCK MANAGEMENT (Cont'd)

6.3.4 Low Power Modes

The user can select an automatic slowdown of clock frequency during Wait for Interrupt operation, thus idling in low power mode while waiting for an interrupt. In WFI operation the clock to the CPU core (CPUCLK) is stopped, thus suspending program execution, while the clock to the peripherals (INTCLK) may be programmed as described in the following paragraphs. Two examples of Low Power operation in WFI are illustrated inFigure 34 and Figure 35.

If low power operation during WFI is disabled (LPOWFI bit = 0 in the CLKCTL Register), the CPU CLK is stopped but INTCLK is unchanged.

If low power operation during Wait for Interrupt is enabled (LPOWFI bit = 1 in the CLKCTL Register), as soon as the CPU executes the WFI instruction, the PLL is turned off and the system clock will be forced to CLOCK2 divided by 16, or to the external low frequency clock, CK_AF, if this has been selected by setting WFI_CKSEL, and providing CKAF_ST is set, thus indicating that the external clock is selected and actually present on the CK_AF pin.

If the external clock source is used, the crystal oscillator may be stopped by setting the XTSTOP bit, providing that the CK_AK clock is present and selected, indicated by CKAF_ST being set. The crystal oscillator will be stopped automatically on entering WFI if the WFI_CKSEL bit has been set. It should be noted that selecting a non-existent CK_AF clock source is impossible, since such a selection requires that the auxiliary clock source be actually present and selected. In no event can a non-existent clock source be selected inadvertently.

It is up to the user program to switch back to a faster clock on the occurrence of an interrupt, taking care to respect the oscillator and PLL stabilisation delays, as appropriate.

It should be noted that any of the low power modes may also be selected explicitly by the user program even when not in Wait for Interrupt mode, by setting the appropriate bits.

6.3.5 Interrupt Generation

System clock selection modifies the CLKCTL and CLK_FLAG registers.

The clock control unit generates an external interrupt request when CK_AF and CLOCK2/16 are selected or deselected as system clock source, as well as when the system clock restarts after a hardware stop (when the STOP MODE feature is available on the specific device). This interrupt can be masked by resetting the INT_SEL bit in the CLKCTL register. Note that this is the only case in the ST9 where an an interrupt is generated with a high to low transition.

Table 13. Summary of Operating Modes using main Crystal Controlled Oscillator

MODE	INTCLK	CPUCLK	DIV2	PRS0-2	CSU_CKSEL	MX1-0	DX2-0	LPOWFI	XT_DIV16
PLL x BY 14	XTAL/2 x (14/D)	INTCLK/N	1	N-1	1	1 0	D-1	X	1
PLL x BY 10	XTAL/2 x (10/D)	INTCLK/N	1	N-1	1	0 0	D-1	X	1
PLL x BY 8	XTAL/2 x (8/D)	INTCLK/N	1	N-1	1	1 1	D-1	X	1
PLL x BY 6	XTAL/2 x (6/D)	INTCLK/N	1	N-1	1	0 1	D-1	X	1
SLOW 1	XTAL/2	INTCLK/N	1	N-1	X	X	111	X	1
SLOW 2	XTAL/32	INTCLK/N	1	N-1	X	X	X	X	0
WAIT FOR INTERRUPT	If LPOWFI=0, no changes occur on INTCLK, but CPUCLK is stopped anyway.								
LOW POWER WAIT FOR INTERRUPT	XTAL/32	STOP	1	X	X	X	X	1	1
RESET	XTAL/2	INTCLK	1	0	0	00	111	0	1
EXAMPLE XTAL=4.4 MHz	2.2*10/2 = 11MHz	11MHz	1	0	1	00	001	X	1

Figure 34. Example of Low Power mode programming in WFI using CK_AF external clock

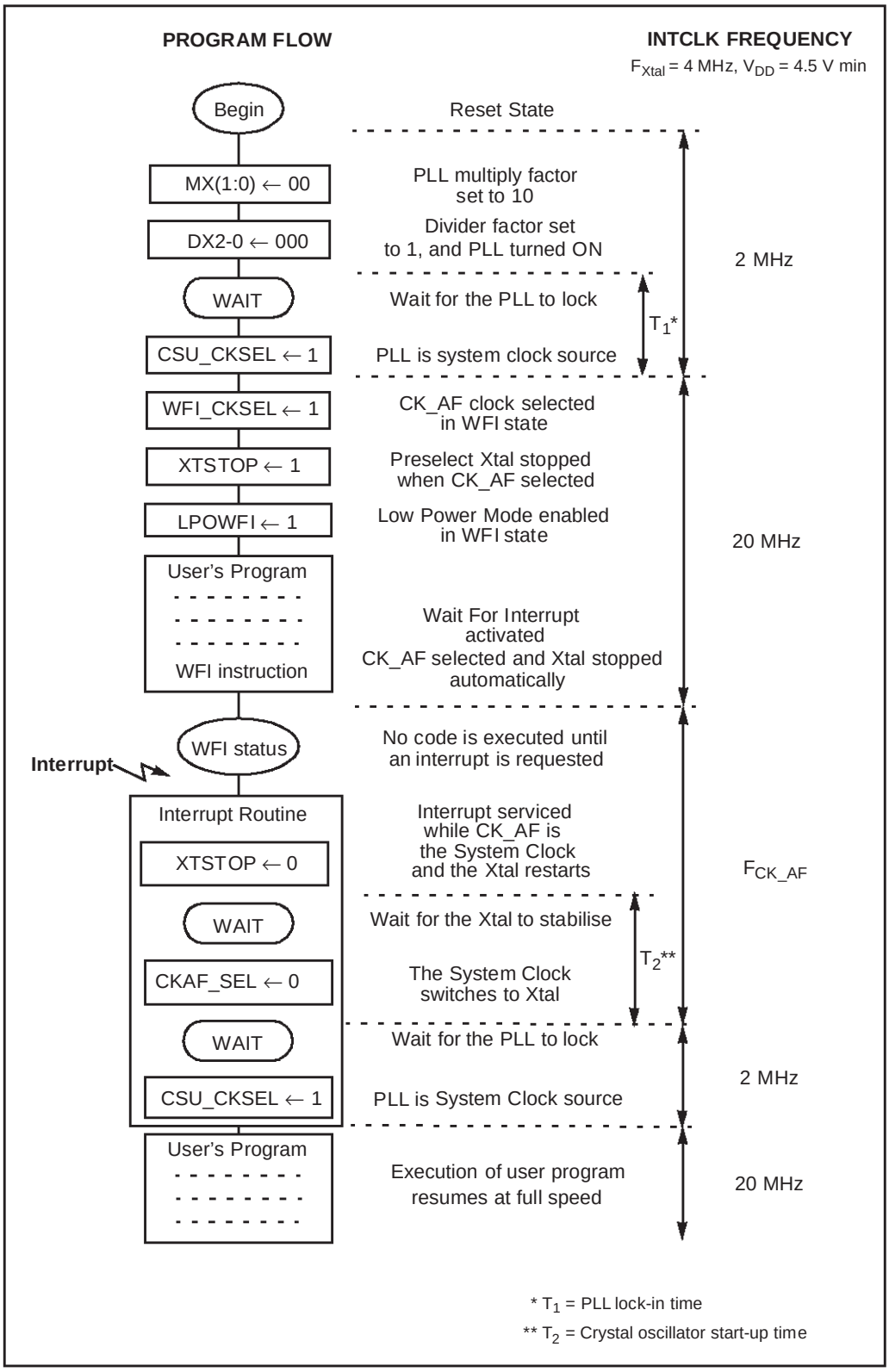
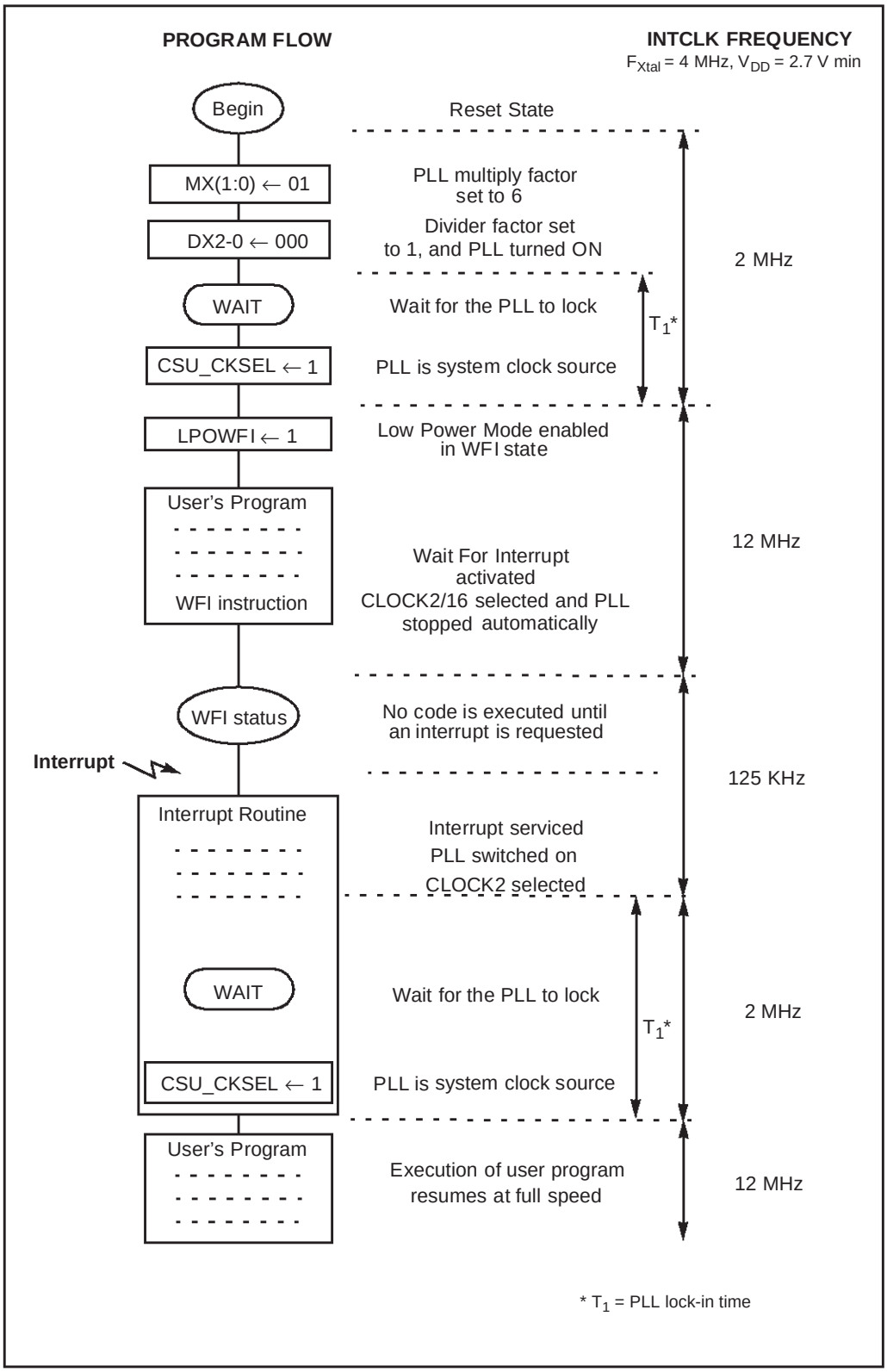


Figure 35. Example of Low Power mode programming in WFI using CLOCK2/16



6.4 CLOCK CONTROL REGISTERS

MODE REGISTER (MODER)
R235 - Read/Write
System Register
Reset Value: 1110 0000 (E0h)

7							0
-	-	DIV2	PRS2	PRS1	PRS0	-	-

***Note**: *This register contains bits which relate to other functions; these are described in the chapter dealing with Device Architecture. Only those bits relating to Clock functions are described here.*

Bit 5 = **DIV2**: *OSCIN Divided by 2*
This bit controls the divide by 2 circuit which operates on the OSCIN Clock.
0: No division of the OSCIN Clock
1: OSCIN clock is internally divided by 2

Bit 4:2 = **PRS[2:0]**: *Clock Prescaling.*
These bits define the prescaler value used to prescale CPUCLK from INTCLK. When these three bits are reset, the CPUCLK is not prescaled, and is equal to INTCLK; in all other cases, the internal clock is prescaled by the value of these three bits plus one.

CLOCK CONTROL REGISTER(CLKCTL)
R240 - Read Write
Register Page: 55
Reset Value: 0000 0000 (00h)

7							0
INT_SEL	-	-	-	SRESEN	CKAF_SEL	WFI_CKSEL	LPOWFI

Bit 7 = **INT_SEL**: *Interrupt Selection.*
0: The external interrupt channel input signal is selected (Reset state)
1: Select the internal RCCU interrupt as the source of the interrupt request

Bit 4:6 = **Reserved for test purposes**
Must be kept reset for normal operation.

Bit 3 = **SRESEN**: *Software Reset Enable.*
0: The HALT instruction turns off the quartz, the PLL and the CCU
1: A Reset is generated when HALT is executed

Bit 2 = **CKAF_SEL**: *Alternate Function Clock Select.*
0: CK_AF clock not selected
1: Select CK_AF clock

Note: To check if the selection has actually occurred, check that CKAF_ST is set. If no clock is present on the CK_AF pin, the selection will not occur.

Bit 1 = **WFI_CKSEL**: *WFI Clock Select.*
This bit selects the clock used in Low power WFI mode if LPOWFI = 1.
0: INTCLK during WFI is CLOCK2/16
1: INTCLK during WFI is CK_AF, providing it is present. In effect this bit sets CKAF_SEL in WFI mode

WARNING: When the CK_AF is selected as Low Power WFI clock but the XTAL is not turned off (R242.4 = 0), after exiting from the WFI, CK_AF will be still selected as system clock. In this case, reset the R240.2 bit to switch back to the XT.

Bit 0 = **LPOWFI**: *Low Power mode during Wait For Interrupt.*
0: Low Power mode during WFI disabled. When WFI is executed, the CPUCLK is stopped and INTCLK is unchanged
1: The ST9 enters Low Power mode when the WFI instruction is executed. The clock during this state depends on WFI_CKSEL

CLOCK CONTROL REGISTERS

CLOCK FLAG REGISTER (CLK_FLAG)

R242 -Read/Write
Register Page: 55
Reset Value: 0100 10x0 after a Watchdog Reset
Reset Value: 0010 10x0 after a Software Reset
Reset Value: 0000 10x0 after a Power-On Reset

7							0
EX_STP	WDGRES	SOFTRES	XTSTOP	XT_DIV16	CKAF_ST	-	CSU_CKSEL

WARNING: If this register is accessed with a logical instruction, such as AND or OR, some bits may not be set as expected.

WARNING: If you select the CK_AF as system clock and turn off the oscillator (bits R240.2 and R242.4 at 1), and then switch back to the XT clock by resetting the R240.2 bit, you must wait for the oscillator to restart correctly (12ms).

Bit 7 = **EX_STP**: *External Stop flag*
This bit is set by hardware and cleared by software.
0: No External Stop condition occurred
1: External Stop condition occurred

Bit 6 = **WDGRES**: *Watchdog reset flag.*
This bit is read only.
0: No Watchdog reset occurred
1: Watchdog reset occurred

Bit 5 = **SOFTRES**: *Software Reset Flag.*
This bit is read only.
0: No software reset occurred
1: Software reset occurred (HALT instruction)

Bit 4 = **XTSTOP**: *External Stop Enable*
0: External stop disabled
1: The Xtal oscillator will be stopped as soon as the CK_AF clock is present and selected, whether this is done explicitly by the user program, or as a result of WFI, if WFI_CKSEL has previously been set to select the CK_AF clock during WFI.

WARNING: When the program writes '1' to the XTSTOP bit, it will still be read as 0 and is only set when the CK_AF clock is running (CKAF_ST=1). Take care, as any operation such as a subsequent AND with '1' or an OR with '0' to the XTSTOP bit will reset it and the oscillator will not be stopped even if CKAF_ST is subsequently set.

Bit 3 = **XT_DIV16**: *CLOCK/16 Selection*

This bit is set and cleared by software. An interrupt is generated when the bit is toggled.
0: CLOCK2/16 is selected and the PLL is off
1: The input is CLOCK2 (or the PLL output depending on the value of CSU_CKSEL)

WARNING: After this bit is modified from 0 to 1, take care that the PLL lock-in time has elapsed before setting the CSU_CKSEL bit.

Bit 2 = **CKAF_ST**: (Read Only)

If set, indicates that the alternate function clock has been selected. If no clock signal is present on the CK_AF pin, the selection will not occur. If reset, the PLL clock, CLOCK2 or CLOCK2/16 is selected (depending on bit 0).

Bit 0 = **CSU_CKSEL**: *CSU Clock Select*
This bit is set and cleared by software. It is also cleared by hardware when:

– bits DX[2:0] (PLLCONF) are set to 111;

– the quartz is stopped (by hardware or software);

– WFI is executed while the LPOWFI bit is set;

– the XT_DIV16 bit (CLK_FLAG) is forced to '0'.

This prevents the PLL, when not yet locked, from providing an irregular clock. Furthermore, a '0' stored in this bit speeds up the PLL's locking.

0: CLOCK2 provides the system clock
1: The PLL Multiplier provides the system clock.

NOTE: Setting the CKAF_SEL bit overrides any other clock selection. Resetting the XT_DIV16 bit overrides the CSU_CKSEL selection (see Figure 32)

PLL CONFIGURATION REGISTER(PLLCONF)
R246 - Read/Write
Register Page: 55
Reset Value: xx00 x111

7							0
-	-	MX1	MX0	-	DX2	DX1	DX0

Bit 5:4 = **MX[1:0]**: *PLL Multiplication Factor.*
Refer to Table 14 PLL Multiplication Factors for multiplier settings.

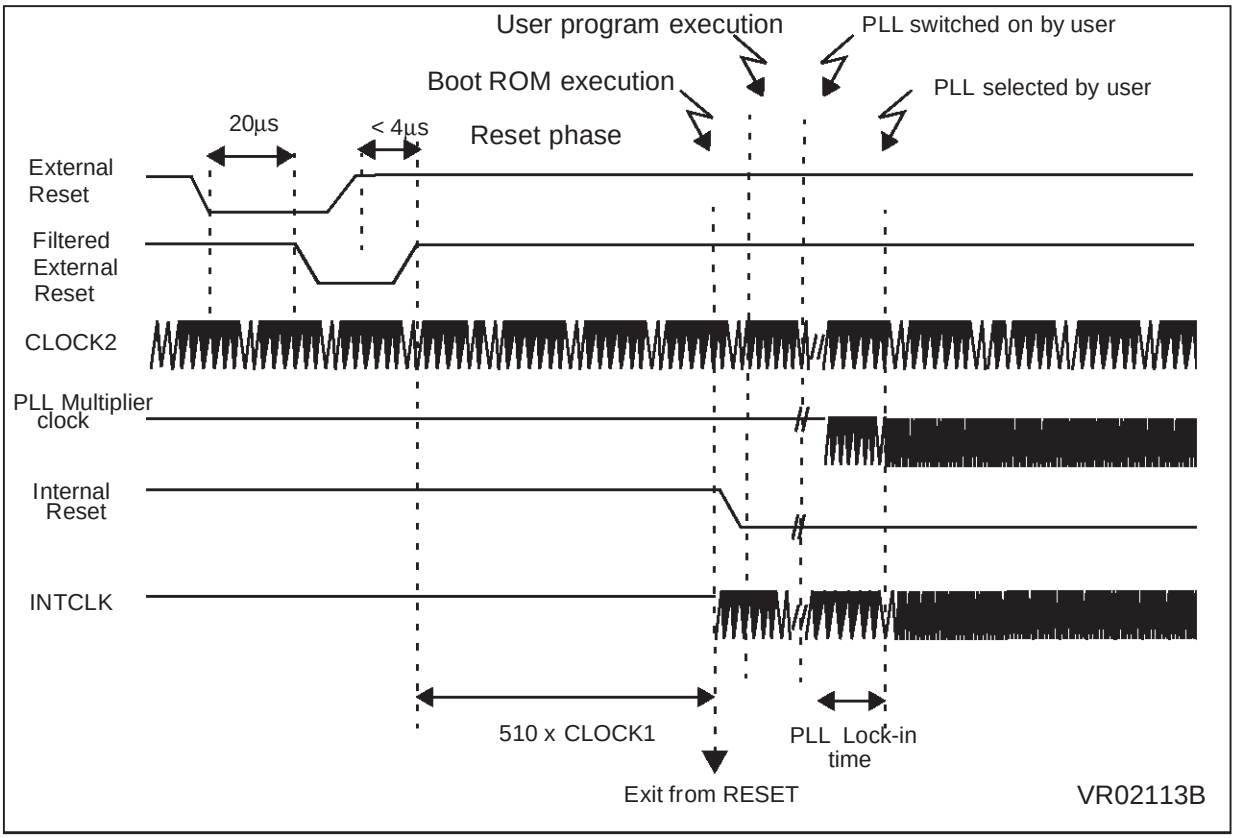
Bit 2:0 = **DX[2:0]**: *PLL output clock divider factor.*
Refer to Table 15 Divider Configuration for divider settings.

MX1	MX0	CLOCK2 x
1	0	14
0	0	10
1	1	8
0	1	6

Table 15. Divider Configuration

DX2	DX1	DX0	CK
0	0	0	PLL CLOCK/1
0	0	1	PLL CLOCK/2
0	1	0	PLL CLOCK/3
0	1	1	PLL CLOCK/4
1	0	0	PLL CLOCK/5
1	0	1	PLL CLOCK/6
1	1	0	PLL CLOCK/7
1	1	1	CLOCK2 (PLL OFF, Reset State)

Figure 36. RCCU General Timing



6.5 OSCILLATOR CHARACTERISTICS

The on-chip oscillator circuit uses an inverting gate circuit with tri-state output.

Notes: *It is recommended to place the quartz or crystal as close as possible to the ST9 to reduce the parasitic capacitance. At low temperature, frost and humidity might prevent from a correct start-up of the oscillator.*

OSCOUT must not be used to drive external circuits.

When the oscillator is stopped, OSCOUT goes high impedance.

In Halt mode, set by means of the HALT instruction, the parallel resistor, R, is disconnected and the oscillator is disabled, forcing the internal clock, CLOCK1, to a high level, and OSCOUT to a high impedance state.

To exit the HALT condition and restart the oscillator, an external RESET pulse is required, having a a minimum duration of 12ms, as illustrated inFigure 40

It should be noted that, if the Watchdog function is enabled, a HALT instruction will not disable the oscillator. This to avoid stopping the Watchdog if a HALT code is executed in error. When this occurs, the CPU will be reset when the Watchdog times out or when an external reset is applied.

Table 16. Oscillator Transconductance

	gm	Min	Typ	Max
5V Operation	mA/V	0.77	1.5	2.4
3V Operation		0.42	0.73	1.47

Figure 37. Crystal Oscillator

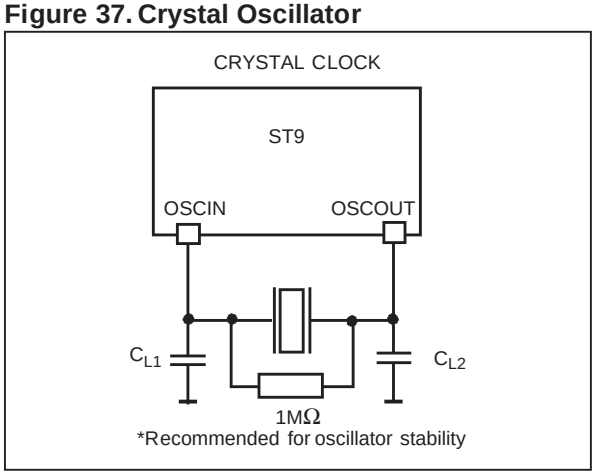


Table 17. Crystal Internal Resistance(Ω) (5V Op.)

$C_1=C_2$ Freq.	56pF	47pF	33pF	22pF
5 Mhz	110	120	210	340
4 Mhz	150	200	330	510
3 Mhz	270	350	560	850

Table 18. Crystal Internal Resistance(Ω) (3V Op.)

$C_1=C_2$ Freq.	56pF	47pF	33pF	22pF
5 Mhz	35	45	75	120
4 Mhz	55	70	125	195
3 Mhz	100	135	220	350

Legend:
C_{L1}, C_{L2}: Maximum Total Capacitance on pins OSCIN and OSCOUT (the value includes the external capacitance tied to the pin CL1 and CL2 plus the parasitic capacitance of the board and of the device).

Note: The tables are relative to the fundamental quartz crystal only (not ceramic resonator).

Figure 38. Internal Oscillator Schematic

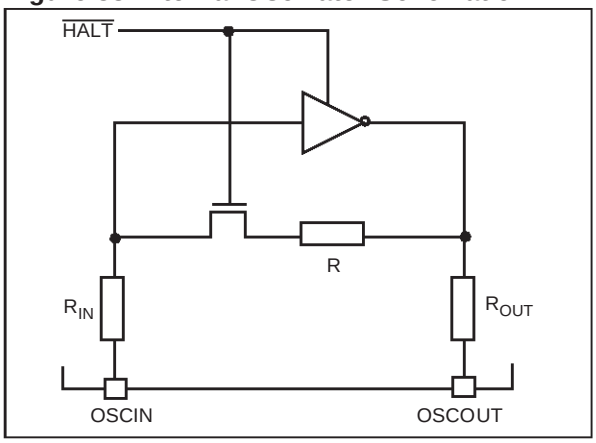
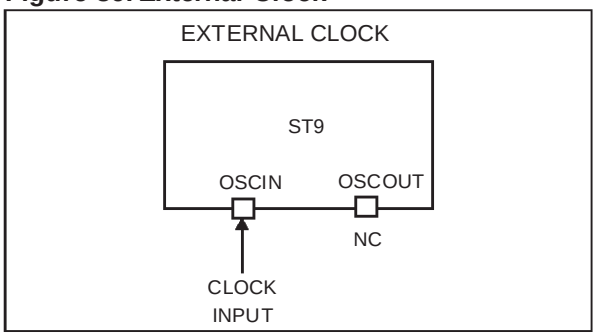


Figure 39. External Clock



6.6 RESET/STOP MANAGER

The Reset/Stop Manager resets the MCU when one of the three following events occurs:

– A Hardware reset, initiated by a low level on the Reset pin.

– A Software reset, initiated by a HALT instruction (when enabled).

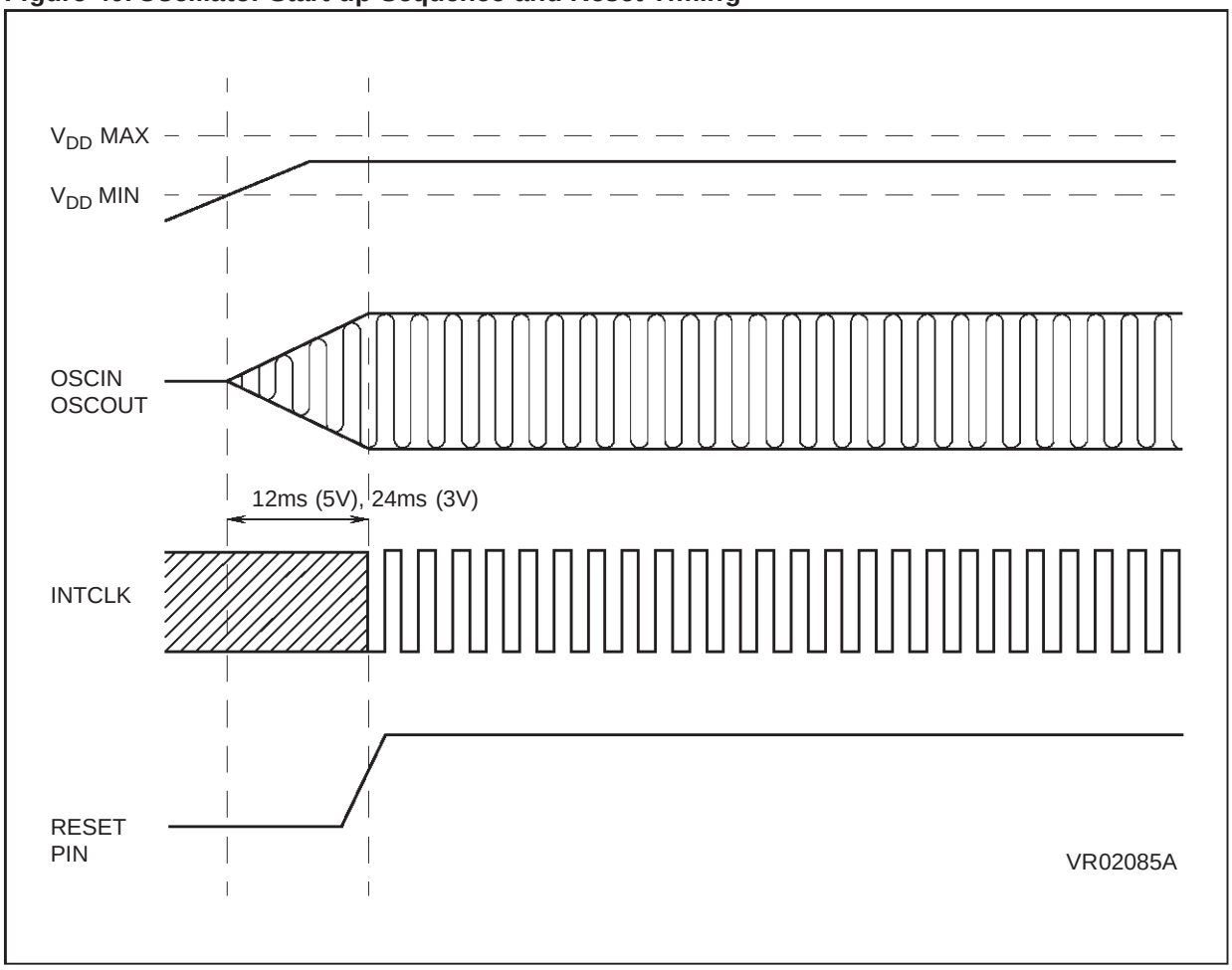
– A Watchdog end of count condition.

The event which caused the last Reset is flagged in the CLK_FLAG register, by setting the SOFTRES or the WDGRES bits respectively; a hardware initiated reset will leave both these bits reset.

The hardware reset overrides all other conditions and forces the ST9 to the reset state. During Reset, the internal registers are set to their reset values, where these are defined, and the I/O pins are set to the Bidirectional Weak Pull-up mode.

Reset is asynchronous: as soon as the reset pin is driven low, a Reset cycle is initiated.

Figure 40. Oscillator Start-up Sequence and Reset Timing



RESET/STOP MANAGER (Cont'd)

The on-chip Timer/Watchdog generates a reset condition if the Watchdog mode is enabled (WCR.WDEN cleared, R252 page 0), and if the programmed period elapses without the specific code (AAh, 55h) written to the appropriate register. The input pin RESET is not driven low by the on-chip reset generated by the Timer/Watchdog.

When the Reset pin goes high again, 510 oscillator clock cycles (CLOCK1) are counted before exiting the Reset state (+-1 CLOCK1 period, depending on the delay between the rising edge of the Reset pin and the first rising edge of CLOCK1). Subsequently a short Boot routine is executed from the device internal Boot ROM, and control then passes to the user program.

The Boot routine sets the device characteristics and loads the correct values in the Memory Management Unit's pointer registers, so that these point to the physical memory areas as mapped in the specific device. The precise duration of this short Boot routine varies from device to device, depending on the Boot ROM contents.

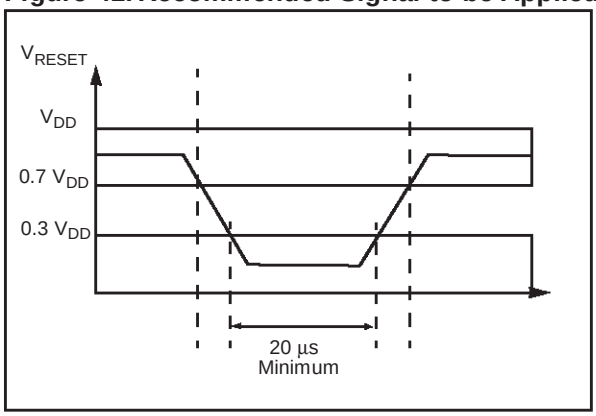
At the end of the Boot routine the Program Counter will be set to the location specified in the Reset Vector located in the lowest two bytes of memory.

6.6.1 Reset Pin Timing

To improve the noise immunity of the device, the Reset pin has a Schmitt trigger input circuit with hysteresis. In addition, a filter will prevent an unwanted reset in case of a single glitch of less than 50 ns on the Reset pin. The device is certain to reset if a negative pulse of more than 20µs is applied. When the reset pin goes high again, a delay of up to 4µs will elapse before the RCCU detects this rising front. From this event on, 510 oscillator clock cycles (CLOCK1) are counted before exiting the Reset state (+-1CLOCK1 period depending on the delay between the positive edge the RCCU detects and the first rising edge of CLOCK1)

If the ST9 is a ROMLESS version, without on-chip program memory, the mermory interface ports are set to external memory mode (i.e Alternate Function) and the memory accesses are made to external Program memory with wait cycles insertion.

Figure 41. Recommended Signal to be Applied on Reset Pin



6.7 EXTERNAL STOP MODE

On ST9 devices provided with an external Stop pin, the Reset/Stop Manager can also stop all oscillators without resetting the device.

To enter Stop Mode, the STOP pin must be forced to “0” for a minimum of 4 system clock cycles; while the Stop pin is kept at “0”, the MCU will remain in Stop Mode and all context information will be preserved. During this condition the internal clock will be frozen in the high state.

When the pin is forced back to “1”, the MCU resumes execution of the user program after a delay of 255 CLOCK2 periods.

On exiting from Stop mode an interrupt is generated and the EX_STP bit in CLK_FLAG will be set, to indicate to the user program that the machine is exiting from Stop mode.

Table 19. Internal Registers Reset Values

Register Number	System Register	Reset Value	Page 0 Register	Reset Value
F	(SSPLR)	undefined	Reserved	
E	(SSPHR)	undefined	(SPICR)	00h
D	(USPLR)	undefined	(SPIDR)	undefined
C	(USPHR)	undefined	(WCR)	7Fh
B	(MODER)	E0h	(WDTCR)	12h
A	(Page Ptr)	undefined	(WDTPR)	undefined
9	(Reg Ptr 1)	undefined	(WDTLR)	undefined
8	(Reg Ptr 0)	undefined	(WDTHR)	undefined
7	(FLAGR)	undefined	(NICR)	00h
6	(CICR)	87h	(EIVR)	x2h
5	(PORT5)	FFh	(EIPLR)	FFh
4	(PORT4)	FFh	(EIMR)	00h
3	(PORT3)	FFh	(EIPR)	00h
2	(PORT2)	FFh	(EITR)	00h
1	(PORT1)	FFh	Reserved	
0	(PORT0)	FFh	Reserved	

7 EXTERNAL MEMORY INTERFACE (EXTMI)

7.1 INTRODUCTION

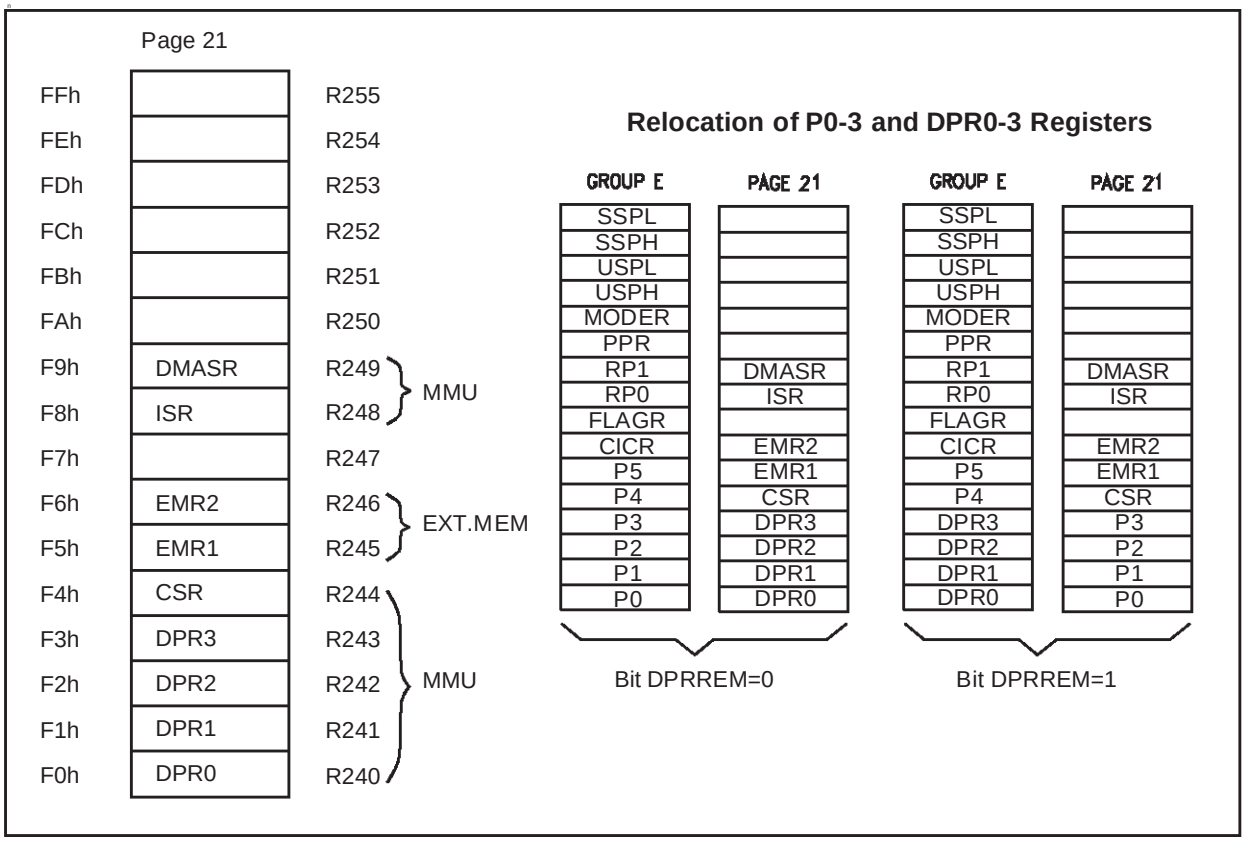
The ST9 External Memory Interface uses two registers (EMR1 and EMR2) to configure external memory accesses. Some interface signals are also affected by WCR - R252 Page 0.

If the two registers EMR1 and EMR2 are set to the proper values, the ST9+ memory access cycle is similar to that of the original ST9, with the only exception that it is composed of just two system clock phases, named T1 and T2.

During phase T1, the memory address is output on the ASN falling edge and is valid on the rising edge of ASN. Port1, Port 2 and Port 6 maintain the address stable until the following T1 phase.

During phase T2, two forms of behavior are possible. If the memory access is a Read cycle, Port 0 pins are released in high-impedance until the next T1 phase and the data signals are sampled by the ST9 on the rising edge of DSN. If the memory access is a Write cycle, on the falling edge of DSN, Port 0 outputs data to be written in the external memory. Those data signals are valid on the rising edge of DSN and are maintained stable until the next address is output. Note that DSN is pulled low at the beginning of phase T2 only during an external memory access.

Figure 42. Page 21 Registers



7.2 EXTERNAL MEMORY SIGNALS

The access to external memory is made using the ASN, DSN, RWN, Port 0, Port1, and WAITN signals described below.

Refer to Figure 43

7.2.1 ASN: Address Strobe

ASN (Output, Active low, Tristate) is active during the System Clock high-level phase of each T1 memory cycle: an ASN rising edge indicates that Memory Address and Read/Write Memory control signals are valid. ASN is released in high-impedance during the bus acknowledge cycle or under the processor control by setting the HIMP bit (MODER.0, R235). Depending on the device ASN is available as Alternate Function or as a dedicated pin.

Under Reset, ASN is held high with an internal weak pull-up.

The behavior of this signal is affected by the MC, ASAF, ETO, BSZ, LAS[1:0] and UAS[1:0] bits in the EMR1 or EMR2 registers. Refer to the Register description.

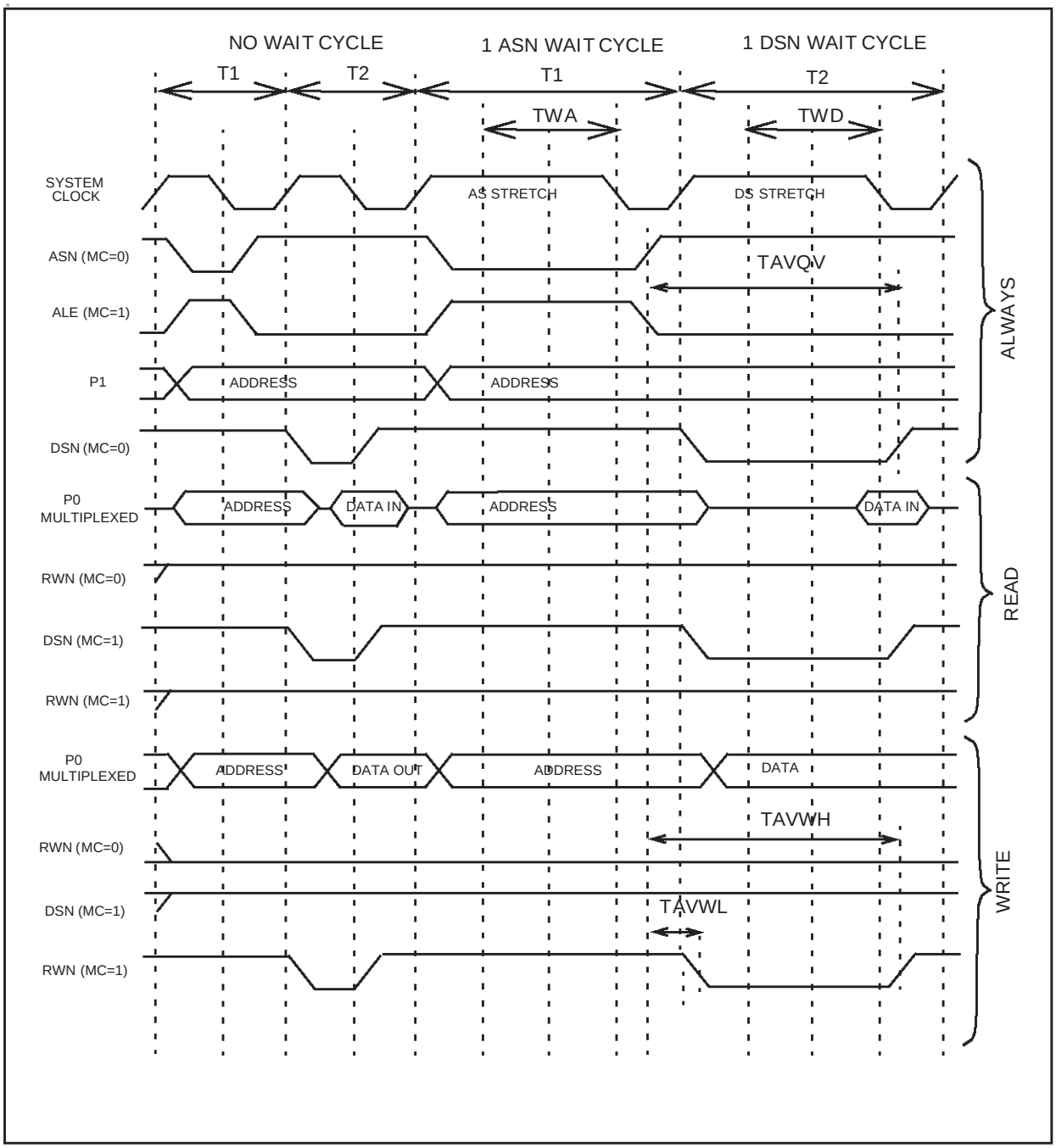
7.2.2 DSN: Data Strobe

DSN (Output, Active low, Tristate) is active during the internal clock high-level phase of each T2 memory cycle. During an external memory read cycle, the data on Port 0 must be valid before the DSN rising edge. During an external memory write cycle, the data on Port 0 are output on the falling edge of DSN and they are valid on the rising edge of DSN. When the internal memory is accessed DSN is kept high during the whole memory cycle. DSN is released in high-impedance during bus acknowledge cycle or under processor control by setting the HIMP bit (MODER.0, R235). Under Reset status, DSN is held high with an internal weak pull-up.

The behavior of this signal is affected by the MC and BSZ bits in the EMR1 register. Refer to the Register description.

EXTERNAL MEMORY SIGNALS(Cont'd)

Figure 43. External memory Read/Write with a programmable wait



EXTERNAL MEMORY SIGNALS(Cont'd)

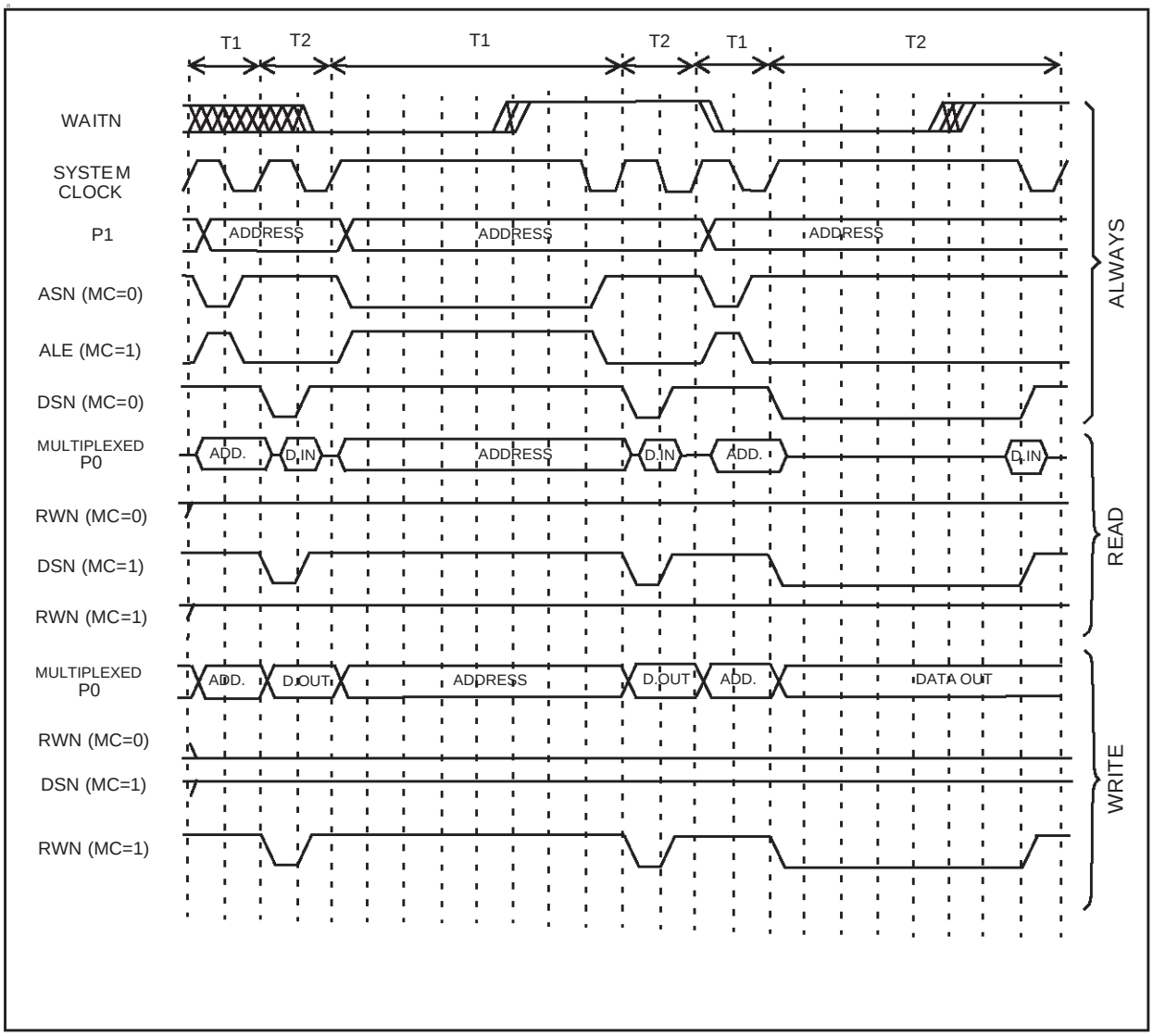
7.2.3 RWN: Read/Write

RWN (Alternate Function Output, Active low, Tristate) identifies the type of memory cycle: RWN="1" identifies a memory read cycle, RWN="0" identifies a memory write cycle. It is defined at the beginning of each memory cycle and it remains stable until the following memory cycle. RWN is released in high-impedance during bus acknowledge cycle or under processor control by setting the HIMP bit (MODER). RWN is enabled via software as the Alternate Function output of the associated I/O port bit (refer to specific ST9 device to identify the port and pin). Under Reset status, the associated bit of the port is set into bidirectional weak pull-up mode.

The behavior of this signal is affected by the MC, ETO and BSZ bits in the EMR1 register. Refer to the Register description.

EXTERNAL MEMORY SIGNALS(Cont'd)

Figure 44. External memory Read/Write sequence with external wait (WAITN pin)



EXTERNAL MEMORY SIGNALS(Cont'd)

7.2.4 PORT 0

If Port 0 (Input/Output, Push-Pull/Open-Drain/ Weak Pull-up) is used as a bit programmable parallel I/O port, it has the same features as a regular port. When set as an Alternate Function, it is used as the External Memory interface: it outputs the multiplexed Address (8 LSB: A7-A0) / Data bus (D7-D0).

7.2.5 PORT 1

If Port 1 (Input/Output, Push-Pull/Open-Drain/ Weak Pull-up) is used as a bit programmable parallel I/O port, it has the same features as a regular port. When set as an Alternate Function, it is used as the external memory interface to provide the 8 MSB of the address (A15-A8).

The behavior of the Port 0 and 1 pins is affected by the BSZ and ETO bits in the EMR1 register. Refer to the Register description.

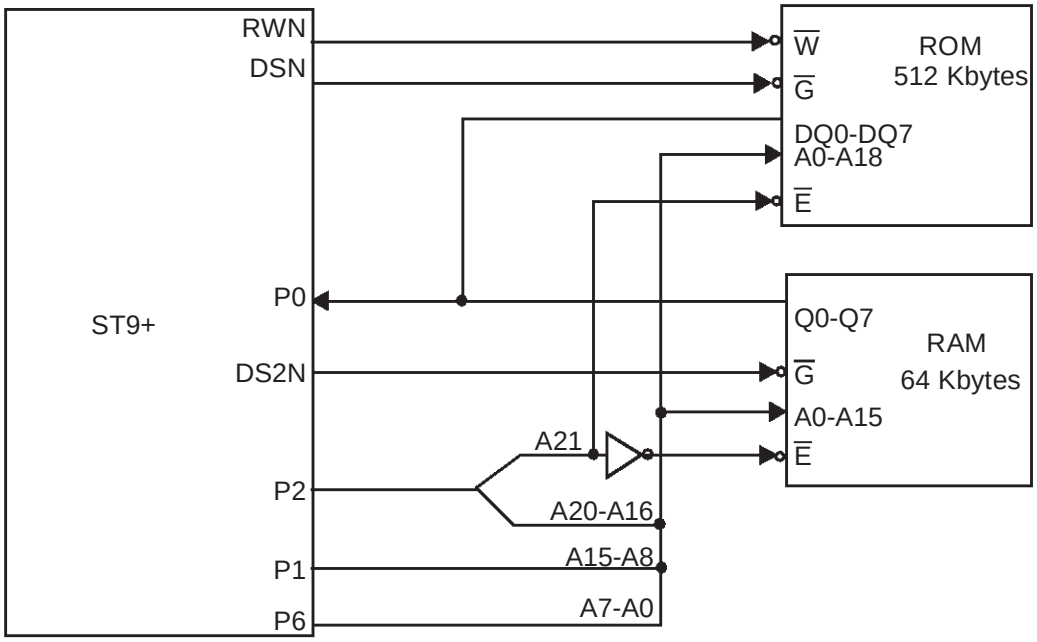
WARNING: The PORT2 available on this device is configured to work as a bit programmable I/O port; it is not used to provide the 6 Most Significant Bits of physical external memory address (bits A21-A16). In this configuration, accessing external memory, the 2 Least Significant Bits of DPR registers are don't care : the 22-bit physical address is given by the concatenation of the 6 MSB of DPR registers and the 16 bits of the virtual address. In other words, the 16-bit virtual address is equal to the 16-bit physical address.

7.2.6 WAITN: External Memory Wait

WAITN (Alternate Function Input, Active low) indicates to the ST9 that the external memory requires more time to complete the memory access cycle. If bit EWEN (EIVR) is set, the WAITN signal is sampled with the rising edge of the processor internal clock during phase T1 or T2 of every memory cycle. If the signal was sampled active, one more internal clock cycle is added to the memory cycle. On the rising edge of the added internal clock cycle, WAITN is sampled again to continue or finish the memory cycle stretching. Note that if WAITN is sampled active during phase T1 then ASN is stretched, while if WAITN is sampled active during phase T2 then DSN is stretched. WAITN is enabled via software as the Alternate Function input of the associated I/O port bit (refer to specific ST9 version to identify the specific port and pin). Under Reset status, the associated bit of the port is set to the bidirectional weak pull-up mode.

Refer to Figure 44

Figure 45. Application Example



7.3 REGISTER DESCRIPTION

EXTERNAL MEMORY REGISTER 1(EMR1)
R245 - Read/Write
Register Page: 21
Reset value: 1000 0000 (80h)

7							0
X	MC	X	ASAF	X	ETO	BSZ	X

Bit 7 = Reserved.

Bit 6 = **MC**: *Mode Control.*
0: ASN, DSN and RWN pins keep the ST9OLD meaning.
1: ASN pin becomes ALE, Address Load Enable (ASN inverted); Thus Memory Adress, Read/ Write signals are valid whenever a falling edge of ALE occurs.
DSN becomes OEN, Output ENable: it keeps the ST9OLD meaning during external read operations, but is forced to “1” during external write operations.
RWN pin becomes WEN, Write ENable: it follows the ST9OLD DSN meaning during external write operations, but is forced to “1” during external read operations.

Bit 5 = Reserved.

Bit 4 = **ASAF**: *Address Strobe as Alternate Function.*
Depending on the device, ASN can be either a dedicated pin or a port Alternate Function. This bit is used only in the second case.
0: ASN Alternate function disabled.
1: ASN Alternate Function enabled.

Bit 2 = **ETO**: *External toggle.*
0: The external memory interface pins (ASN, DSN, RWN, Port0, Port1) toggle only if an access to external memory is performed.
1: When the internal memory protection is disabled (mask option), the above pins (except DSN which never toggles during internal memory accesses) toggle during both internal and external memory accesses.

Bit 1 = **BSZ**: *Bus size.*
0: The external memory interface pins use smaller, less noisy output buffers. This may limit the operation frequency of the device, unless the clock is slow enough or sufficient wait states are inserted.
1: The external memory interface pins (ASN, DSN, R/W, Port 0, 1) use larger, more noisy output buffers .

Bit 0 = Reserved.

***WARNING**: External memory must be correctly addressed before and after a write operation on the EMR1 register. For example, if code is fetched from external memory using the ST9OLD external memory interface configuration (MC=0), setting the MC bit will cause the device to behave unpredictably.*

EXTERNAL MEMORY INTERFACE REGISTERS(Cont'd)

EXTERNAL MEMORY REGISTER 2 (EMR2)
R246 - Read/Write
Register Page: 21
Reset value: 0000 1111 (0Fh)

7							0
-	ENCSR	DPRREM	0	LAS1	LAS0	UAS1	UAS0

Bit 7 = Reserved.

Bit 6 = **ENCSR**: *Enable Code Segment Register.*

This bit affects the ST9 CPU behavior whenever an interrupt request is issued.

0: The CPU works in original ST9 compatibility mode concerning stack frame during interrupts. For the duration of the interrupt service routine, ISR is used instead of CSR, and the interrupt stack frame is identical to that of the original ST9: only the PC and Flags are pushed. This avoids saving the CSR on the stack in the event of an interrupt, thus ensuring a faster interrupt response time. The drawback is that it is not possible for an interrupt service routine to perform inter-segment calls or jumps: these instructions would update the CSR, which, in this case, is not used (ISR is used instead). The code segment size for all interrupt service routines is thus limited to 64K bytes.

1: If ENCSR is set, ISR is only used to point to the interrupt vector table and to initialize the CSR at the beginning of the interrupt service routine: the old CSR is pushed onto the stack together with the PC and flags, and CSR is then loaded with the contents of ISR. In this case, iret will also restore CSR from the stack. This approach allows interrupt service routines to access the entire 4Mbytes of address space; the drawback is that the interrupt response time is slightly increased, because of the need to also save CSR on the stack. Full compatibility with the original ST9 is lost in this case, because the interrupt stack frame is different; this difference, however, should not affect the vast majority of programs.

Bit 5 = **DPRREM**: *Data Page Registers remapping*

0: The locations of the four MMU (Memory Management Unit) Data Page Registers (DPR0, DPR1, DPR2 and DPR3) are in page 21.

1: The four MMU Data Page Registers are swapped with that of the Data Registers of ports 0-3.

Refer to Figure 42

Bit 4 = Warning : Must be set by the user when using the external memory interface (Reset value is 0).

Bit 3:2 = **LAS[1:0]**: *Lower memory address strobe stretch*.

These two bits contain the number of wait cycles (from 0 to 3) to add to the System Clock to stretch ASN during external lower memory block accesses (A21="0"). The reset value is 3.

EXTERNAL MEMORY INTERFACE REGISTERS(Cont'd)

Bit 1:0 = **UAS[1:0]**: *Upper memory address strobe stretch*.
These two bits contain the number of wait cycles (from 0 to 3) to add to the System Clock to stretch ASN during external upper memory block accesses (A21=1) . The reset value is 3.

WARNING: The EMR2 register cannot be written during an interrupt service routine.

WAIT CONTROL REGISTER (WCR)
R252 - Read/Write
Register Page: 0
Reset Value: 0111 1111 (7Fh)

7							0
0	WDGEN	UDS2	UDS1	UDS0	LDS2	LDS1	LDS0

Bit 7 = Reserved, forced by hardware to 0.

Bit 6 = **WDGEN**: *Watchdog Enable.*

For a description of this bit, refer to the Timer/ Watchdog chapter.

WARNING: Clearing this bit has the effect of setting the Timer/Watchdog to Watchdog mode. Unless this is desired, it must be set to "1".

Bit 5:3 = **UDS[2:0]**: *Upper memory data strobe stretch.*
These bits contain the number of INTCLK cycles to be added automatically to DSN for external upper memory block accesses. UDS = 0 adds no additional wait cycles. UDS = 7 adds the maximum 7 INTCLK cycles (reset condition).

Bit 2:0 = **LDS[2:0]**: *Lower memory data strobe stretch.*
These bits contain the number of INTCLK cycles to be added automatically to DSN for external lower memory block accesses. LDS = 0 adds no additional wait cycles, LDS = 7 adds the maximum 7 INTCLK cycles (reset condition).

Note 1: The number of clock cycles added refers to INTCLK and NOT to CPUCLK.

Note 2: The distinction between the Upper memory block and the Lower memory block allows different wait cycles between the first 2 Mbytes and the second 2 Mbytes, and allows 2 different data strobe signals to be used to access 2 different memories.

Typically, the RAM will be located above address 0x200000 and the ROM below address 0x1FFFFF, with different access times. No extra hardware is required as DSN is used to access the upper memory block and DS2N is used to access the lower memory block.

WARNING: The reset value of the Wait Control Register gives the maximum number of Wait cycles for external memory. To get optimum performance from the ST9, the user should write the UDS[2:0] and LDS[2:0] bits to 0, if the external addressed memories are fast enough.

8 I/O PORTS

8.1 INTRODUCTION

ST9 devices feature flexible individually programmable multifunctional input/output lines. Refer to the Pin Description Chapter for specific pin allocations. These lines, which are logically grouped as 8-bit ports, can be individually programmed to provide digital input/output and analog input, or to connect input/output signals to the on-chip peripherals as alternate pin functions. All ports can be individually configured as an input, bi-directional, output or alternate function. In addition, pull-ups can be turned off for open-drain operation, and weak pull-ups can be turned on in their place, to avoid the need for off-chip resistive pull-ups. Ports configured as open drain must never have voltage on the port pin exceeding V_{DD} (refer to the Electrical Characteristics section). Input buffers can be either TTL or CMOS compatible. Alternatively some input buffers can be permanently forced by hardware to operate as Schmitt triggers.

8.2 SPECIFIC PORT CONFIGURATIONS

Refer to the Pin Description chapter for a list of the specific port styles and reset values.

8.3 PORT CONTROL REGISTERS

Each port is associated with a Data register (PxDR) and three Control registers (PxC0, PxC1, PxC2). These define the port configuration and allow dynamic configuration changes during program execution. Port Data and Control registers are mapped into the Register File as shown inFigure 46. Port Data and Control registers are treated just like any other general purpose register. There are no special instructions for port manipulation: any instruction that can address a register, can address the ports. Data can be directly accessed in the port register, without passing through other memory or “accumulator” locations.

Figure 46. I/O Register Map

	GROUP E			GROUP F PAGE 2	GROUP F PAGE 3	GROUP F PAGE 43	
	System Registers		FFh	Reserved	P7DR	P9DR	R255
			FEh	P3C2	P7C2	P9C2	R254
			FDh	P3C1	P7C1	P9C1	R253
			FCh	P3C0	P7C0	P9C0	R252
			FBh	Reserved	P6DR	P8DR	R251
			FAh	P2C2	P6C2	P8C2	R250
			F9h	P2C1	P6C1	P8C1	R249
			F8h	P2C0	P6C0	P8C0	R248
			F7h	Reserved	Reserved	Reserved	R247
			F6h	P1C2	P5C2		R246
E5h	P5DR	R229	F5h	P1C1	P5C1		R245
E4h	P4DR	R228	F4h	P1C0	P5C0		R244
E3h	P3DR	R227	F3h	Reserved	Reserved		R243
E2h	P2DR	R226	F2h	P0C2	P4C2		R242
E1h	P1DR	R225	F1h	P0C1	P4C1		R241
E0h	P0DR	R224	F0h	P0C0	P4C0		R240

CONTROL REGISTERS (Cont'd)

During Reset, ports with weak pull-ups are set in bidirectional/weak pull-up mode and the output Data Register is set to FFh. This condition is also held after Reset, except for Ports 0 and 1 in ROM-less devices, and can be redefined under software control.

Bidirectional ports without weak pull-ups are set in high impedance during reset. To ensure proper levels during reset, these ports must be externally connected to either V_{DD} or V_{SS} through external pull-up or pull-down resistors.

Other reset conditions may apply in specific ST9 devices.

8.4 INPUT/OUTPUT BIT CONFIGURATION

By programming the control bits PxC0.n and PxC1.n (see Figure 47) it is possible to configure bit Px.n as Input, Output, Bidirectional or Alternate Function Output, where X is the number of the I/O port, and n the bit within the port (n = 0 to 7).

When programmed as input, it is possible to select the input level as TTL or CMOS compatible by programming the relevant PxC2.n control bit, except where the Schmitt trigger option is assigned to the pin.

The output buffer can be programmed as push-pull or open-drain.

A weak pull-up configuration can be used to avoid external pull-ups when programmed as bidirectional (except where the weak pull-up option has been permanently disabled in the pin hardware assignment).

Each pin of an I/O port may assume software programmable Alternate Functions (refer to the device Pin Description and to Section 8.5 ALTERNATE FUNCTION ARCHITECTURE). To output signals from the ST9 peripherals, the port must be configured as AF OUT. On ST9 devices with A/D Converter(s), configure the ports used for analog inputs as AF IN.

The basic structure of the bit Px.n of a general purpose port Px is shown in Figure 48.

Independently of the chosen configuration, when the user addresses the port as the destination register of an instruction, the port is written to and the data is transferred from the internal Data Bus to the Output Master Latches. When the port is addressed as the source register of an instruction, the port is read and the data (stored in the Input Latch) is transferred to the internal Data Bus.

When Px.n is programmed as an Input
(See Figure 49).

– The Output Buffer is forced tristate.

– The data present on the I/O pin is sampled into the Input Latch at the beginning of each instruction execution.

– The data stored in the Output Master Latch is copied into the Output Slave Latch at the end of the execution of each instruction. Thus, if bit Px.n is reconfigured as an Output or Bidirectional, the data stored in the Output Slave Latch will be reflected on the I/O pin.

INPUT/OUTPUT BIT CONFIGURATION(Cont'd)

Figure 47. Control Bits

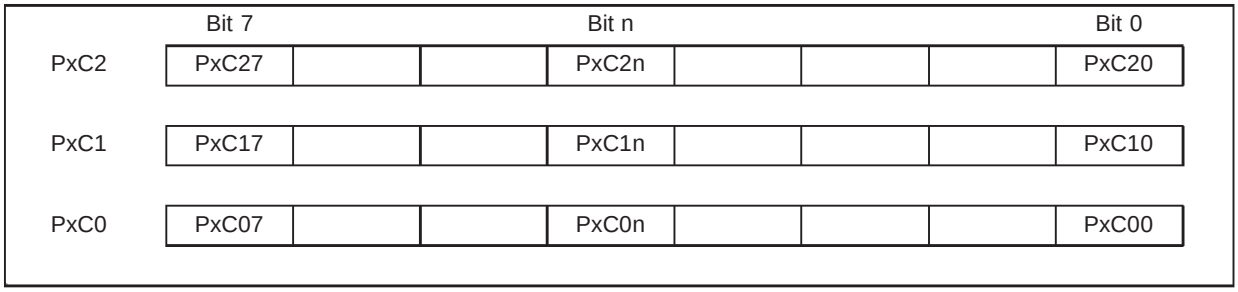


Table 20. Port Bit Configuration Table (n = 0, 1... 7; X = port number)

	General Purpose I/O Pins								ADC Pins
PXC2n	0	1	0	1	0	1	0	1	1
PXC1n	0	0	1	1	0	0	1	1	1
PXC0n	0	0	0	0	1	1	1	1	1
PXn Configuration	BID(1)	BID	OUT	OUT	IN	IN	AF OUT	AF OUT	AF IN
PXn Output Type	WP OD	OD	PP	OD	HI-Z	HI-Z	PP	OD	HI-Z(2)
PXn Input Type	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	CMOS (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	TTL (or Schmitt Trigger)	Analog Input

(1) Reset state of standard I/O port bit with weak pull-up.

(2) For A/D Converter inputs.

Legend:

X = Port
n = Bit
AF = Alternate Function
BID = Bidirectional
CMOS= CMOS Standard Input Levels
HI-Z = High Impedance
IN = Input
OD = Open Drain
OUT = Output
PP = Push-Pull
TTL = TTL Standard Input Levels
WP = Weak Pull-up

INPUT/OUTPUT BIT CONFIGURATION(Cont'd)

Figure 48. Basic Structure of an I/O Port Pin

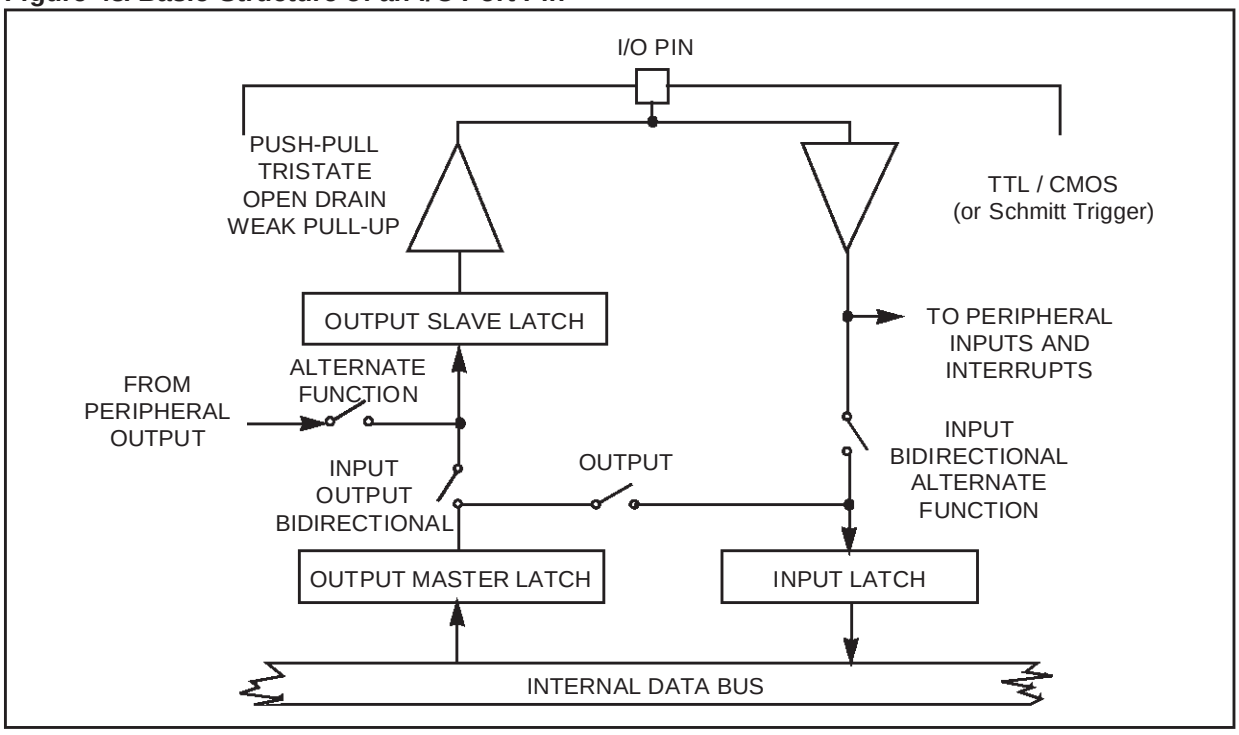


Figure 49. Input Configuration

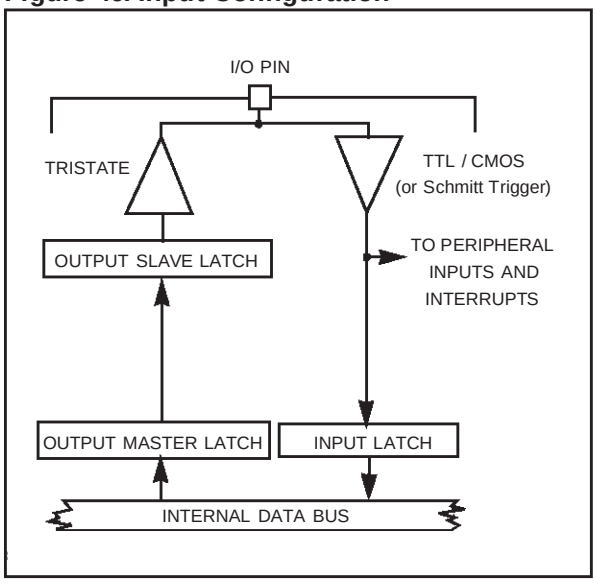
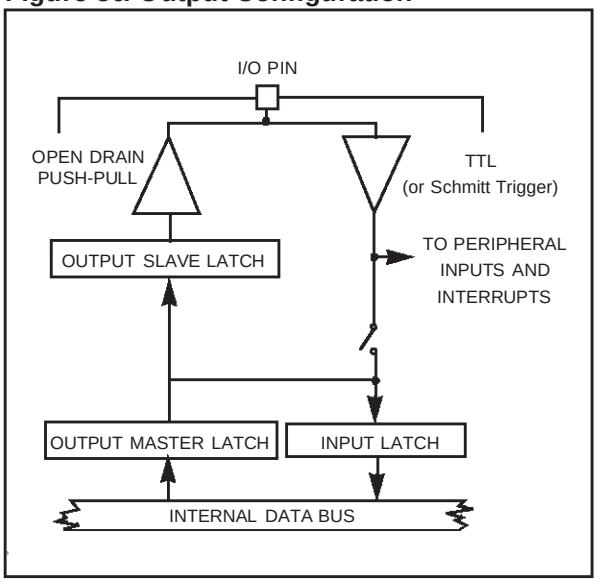


Figure 50. Output Configuration



INPUT/OUTPUT BIT CONFIGURATION(Cont'd)

When Px.n is programmed as an Output
(Figure 50)

– The Output Buffer is turned on in an Open-drain or Push-pull configuration.

– The data stored in the Output Master Latch is copied both into the Input Latch and into the Output Slave Latch, driving the I/O pin, at the end of the execution of the instruction.

When Px.n is programmed as Bidirectional
(Figure 51)

– The Output Buffer is turned on in an Open-Drain or Weak Pull-up configuration (except when disabled in hardware).

– The data present on the I/O pin is sampled into the Input Latch at the beginning of the execution of the instruction.

– The data stored in the Output Master Latch is copied into the Output Slave Latch, driving the I/O pin, at the end of the execution of the instruction.

WARNING: Due to the fact that in bidirectional mode the external pin is read instead of the output latch, particular care must be taken with arithmetic/logic and Boolean instructions performed on a bidirectional port pin.

These instructions use a read-modify-write sequence, and the result written in the port register depends on the logical level present on the external pin.

This may bring unwanted modifications to the port output register content.

For example:

Port register content, 0Fh
external port value, 03h
(Bits 3 and 2 are externally forced to 0)

A `bset` instruction on bit 7 will return:

Port register content, 83h
external port value, 83h
(Bits 3 and 2 have been cleared).

To avoid this situation, it is suggested that all operations on a port, using at least one bit in bidirectional mode, are performed on a copy of the port register, then transferring the result with a load instruction to the I/O port.

When Px.n is programmed as a digital Alternate Function Output
(Figure 52)

– The Output Buffer is turned on in an Open-Drain or Push-Pull configuration.

– The data present on the I/O pin is sampled into the Input Latch at the beginning of the execution of the instruction.

– The signal from an on-chip function is allowed to load the Output Slave Latch driving the I/O pin. Signal timing is under control of the alternate function. If no alternate function is connected to Px.n, the I/O pin is driven to a high level when in Push-Pull configuration, and to a high impedance state when in open drain configuration.

Figure 51. Bidirectional Configuration

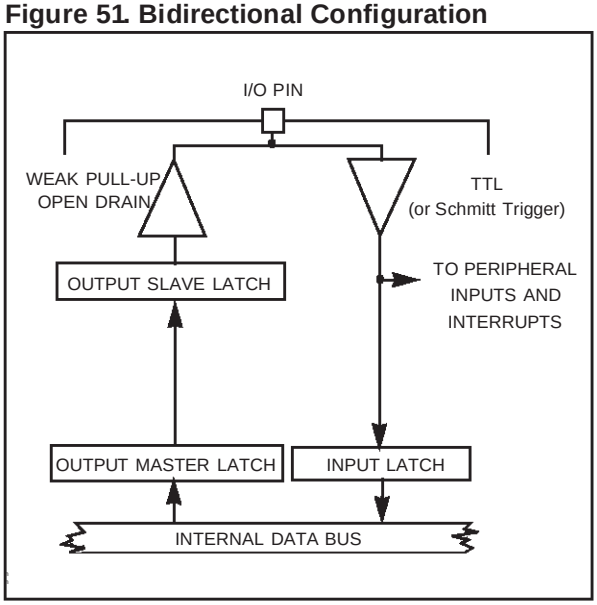
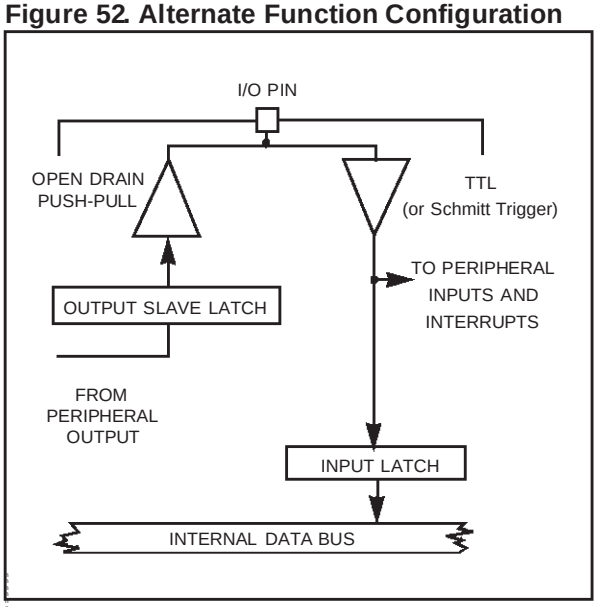


Figure 52. Alternate Function Configuration



8.5 ALTERNATE FUNCTION ARCHITECTURE

Each I/O pin may be connected to three different types of internal signal:

– Data bus Input/Output

– Alternate Function Input

– Alternate Function Output

8.5.1 Pin Declared as I/O

A pin declared as I/O, is connected to the I/O buffer. This pin may be an Input, an Output, or a bidirectional I/O, depending on the value stored in (PxC2, PxC1 and PxC0).

8.5.2 Pin Declared as an Alternate Input

A single pin may be directly connected to several Alternate inputs. In this case, the user must select the required input mode (with the PxC2, PxC1, PxC0 bits) and enable the selected Alternate Function in the Control Register of the peripheral. No specific port configuration is required to enable an Alternate Function input, since the input buffer is directly connected to each alternate function module on the shared pin. As more than one module can use the same input, it is up to the user software to enable the required module as necessary. Parallel I/Os remain operational even when using an Alternate Function input. The exception to this is when an I/O port bit is permanently assigned by hardware as an A/D bit. In this case , after software programming of the bit in AF-OD-TTL, the Alternate function output is forced to logic level 1. The analog voltage level on the corresponding pin is directly input to the ADC.

8.5.3 Pin Declared as an Alternate Function Output

The user must select the AF OUT configuration using the PxC2, PxC1, PxC0 bits. Several Alternate Function outputs may drive a common pin. In such case, the Alternate Function output signals are logically ANDed before driving the common pin. The user must therefore enable the required Alternate Function Output by software.

***WARNING**: When a pin is connected both to an alternate function output and to an alternate function input, it should be noted that the output signal will always be present on the alternate function input.*

8.6 I/O STATUS AFTER WFI, HALT AND RESET

The status of the I/O ports during the Wait For Interrupt, Halt and Reset operational modes is shown in the following table. The External Memory Interface ports are shown separately. If only the internal memory is being used and the ports are acting as I/O, the status is the same as shown for the other I/O ports.

Mode	Ext. Mem - I/O Ports		I/O Ports
	P0	P1, P2, P6	
WFI	High Impedance or next address (depending on the last memory operation performed on Port)	Next Address	Not Affected (clock outputs running)
HALT	High Impedance	Next Address	Not Affected (clock outputs stopped)
RESET	Alternate function push-pull (ROMless device)		Bidirectional Weak Pull-up (High impedance when disabled in hardware).

9 ON-CHIP PERIPHERALS

9.1 TIMER/WATCHDOG (WDT)

9.1.1 Introduction

The Timer/Watchdog (WDT) peripheral consists of a programmable 16-bit timer and an 8-bit prescaler. It can be used, for example, to:

– Generate periodic interrupts

– Measure input signal pulse widths

– Request an interrupt after a set number of events

– Generate an output signal waveform

– Act as a Watchdog timer to monitor system integrity

The main WDT registers are:

– Control register for the input, output and interrupt logic blocks (WDTCR)

– 16-bit counter register pair (WDTHR, WDTLR)

– Prescaler register (WDTPR)

The hardware interface consists of up to five signals:

– WDIN External clock input

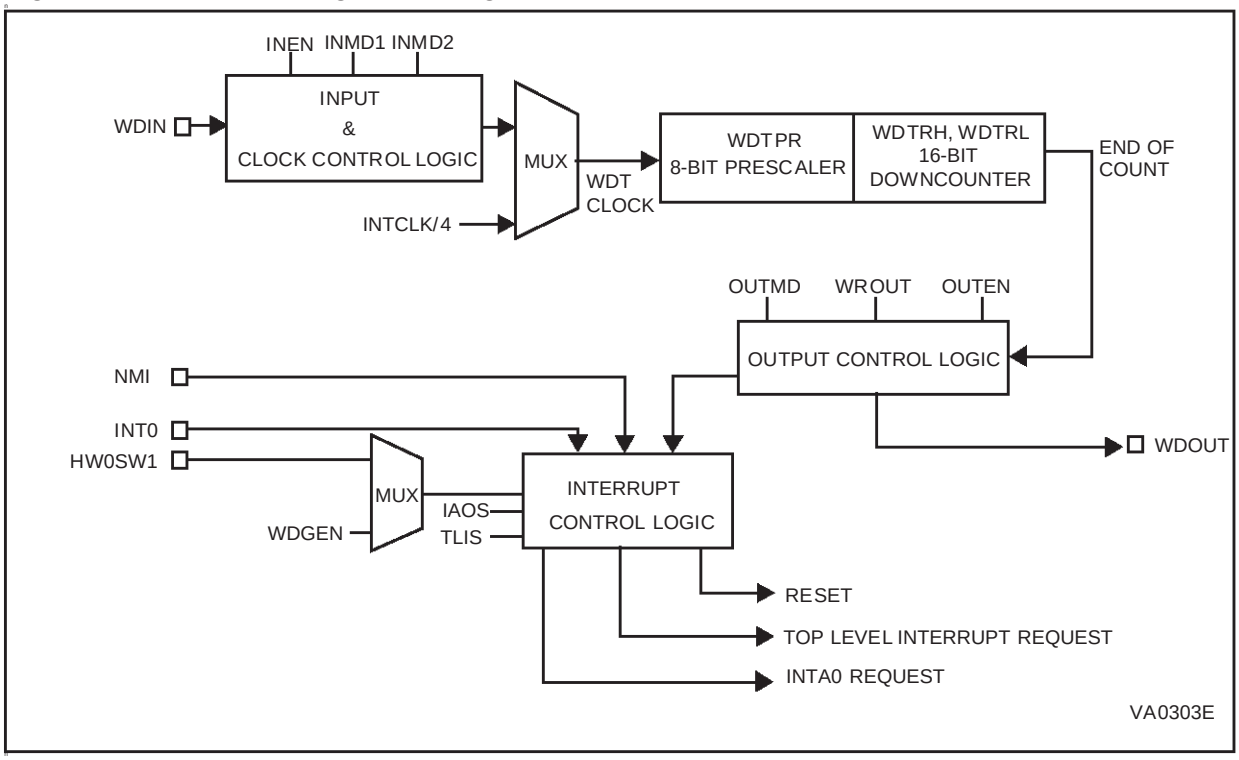
– WDOUT Square wave or PWM signal output

– INT0 External interrupt input

– NMI Non-Maskable Interrupt input

– HW0SW1 Hardware/Software Watchdog enable

Figure 53. Timer/Watchdog Block Diagram



9.1.1.1 Device-specific Options

Depending on the ST9 variant and package type, some WDT interface signals described in this chapter may not be connected to an external pin. Refer to the device pinout description.

TIMER/WATCHDOG (Cont'd)

9.1.2 Functional Description

9.1.2.1 External Signals

The HW0SW1 pin can be used to permanently enable Watchdog mode. Refer to section 9.1.3.1 on page 98.

The WDIN Input pin can be used in one of four modes:

– Event Counter Mode

– Gated External Input Mode

– Triggerable Input Mode

– Retriggerable Input Mode

The WDOUT output pin can be used to generate a square wave or a Pulse Width Modulated signal.

An interrupt, generated when the WDT is running as the 16-bit Timer/Counter, can be used as a Top Level Interrupt or as an interrupt source connected to channel A0 of the external interrupt structure (replacing the INT0 interrupt input).

The counter can be driven either by an external clock, or internally by INTCLK divided by 4.

9.1.2.2 Initialisation

The prescaler (WDTPR) and counter (WDTRL, WDTRH) registers must be loaded with initial values before starting the Timer/Counter. If this is not done, counting will start with reset values.

9.1.2.3 Start/Stop

The ST_SP bit enables downcounting. When this bit is set, the Timer will start at the beginning of the following instruction. Resetting this bit stops the counter.

If the counter is stopped and restarted, counting will resume from the last value unless a new constant has been entered in the Timer registers (WDTRL, WDTRH).

A new constant can be written in the WDTRH, WDTRL, WDTPR registers while the counter is running. The new value of the WDTRH, WDTRL registers will be loaded at the next End of Count (EOC) condition while the new value of the WDTPR register will be effective immediately.

End of Count is when the counter is 0.

When Watchdog mode is enabled the state of the ST_SP bit is irrelevant.

9.1.2.4 Single/Continuous Mode

The S_C bit allows selection of single or continuous mode.This Mode bit can be written with the Timer stopped or running. It is possible to toggle the S_C bit and start the counter with the same instruction.

Single Mode

On reaching the End Of Count condition, the Timer stops, reloads the constant, and resets the Start/Stop bit. Software can check the current status by reading this bit. To restart the Timer, set the Start/Stop bit.

Note: If the Timer constant has been modified during the stop period, it is reloaded at start time.

Continuous Mode

On reaching the End Of Count condition, the counter automatically reloads the constant and restarts. It is stopped only if the Start/Stop bit is reset.

9.1.2.5 Input Section

If the Timer/Counter input is enabled (INEN bit) it can count pulses input on the WDIN pin. Otherwise it counts the internal clock/4.

For instance, when INTCLK = 20MHz, the End Of Count rate is:

3.35 seconds for Maximum Count
(Timer Const. = FFFFh, Prescaler Const. = FFh)

200 ns for Minimum Count
(Timer Const. = 0000h, Prescaler Const. = 00h)

The Input pin can be used in one of four modes:

– Event Counter Mode

– Gated External Input Mode

– Triggerable Input Mode

– Retriggerable Input Mode

The mode is configurable in the WDTCR.

9.1.2.6 Event Counter Mode

In this mode the Timer is driven by the external clock applied to the input pin, thus operating as an event counter. The event is defined as a high to low transition of the input signal. Spacing between trailing edges should be at least 8 INTCLK periods (or 400ns with INTCLK = 20MHz).

Counting starts at the next input event after the ST_SP bit is set and stops when the ST_SP bit is reset.

TIMER/WATCHDOG (Cont'd)

9.1.2.7 Gated Input Mode

This mode can be used for pulse width measurement. The Timer is clocked by INTCLK/4, and is started and stopped by means of the input pin and the ST_SP bit. When the input pin is high, the Timer counts. When it is low, counting stops. The maximum input pin frequency is equivalent to INTCLK/8.

9.1.2.8 Triggerable Input Mode

The Timer (clocked internally by INTCLK/4) is started by the following sequence:

– setting the Start-Stop bit, followed by

– a High to Low transition on the input pin.

To stop the Timer, reset the ST_SP bit.

9.1.2.9 Retriggerable Input Mode

In this mode, the Timer (clocked internally by INTCLK/4) is started by setting the ST_SP bit. A High to Low transition on the input pin causes counting to restart from the initial value. When the Timer is stopped (ST_SP bit reset), a High to Low transition of the input pin has no effect.

9.1.2.10 Timer/Counter Output Modes

Output modes are selected by means of the OUTEN (Output Enable) and OUTMD (Output Mode) bits of the WDTCR register.

No Output Mode
(OUTEN = "0")

The output is disabled and the output pin is set high, in order to allow alternate I/O pin functions.

Square Wave Output Mode
(OUTEN = "1", OUTMD = "0")

The Timer outputs a signal with a frequency equal to half the End of Count repetition rate on the WDOUT pin. With an INTCLK frequency of 20MHz, this allows a square wave signal to be generated whose period can range from 400ns to 6.7 seconds.

Pulse Width Modulated Output Mode
(OUTEN = "1", OUTMD = "1")

The state of the WROUT bit is transferred to the output pin (WDOUT) at the End of Count, and is held until the next End of Count condition. The user can thus generate PWM signals by modifying the status of the WROUT pin between End of Count events, based on software counters decremented by the Timer Watchdog interrupt.

9.1.3 Watchdog Timer Operation

This mode is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence of operation. The Watchdog, when enabled, resets the MCU, unless the program executes the correct write sequence before expiry of the programmed time period. The application program must be designed so as to correctly write to the WDTLR Watchdog register at regular intervals during all phases of normal operation.

9.1.3.1 Hardware Watchdog/Software Watchdog

The HW0SW1 pin (when available) selects Hardware Watchdog or Software Watchdog.

If HW0SW1 is held low:

– The Watchdog is enabled by hardware immediately after an external reset. (Note: Software reset or Watchdog reset have no effect on the Watchdog enable status).

– The initial counter value (FFFFh) cannot be modified, however software can change the prescaler value on the fly.

– The WDGEN bit has no effect. (Note: it is not forced low).

If HW0SW1 is held high, or is not present:

– The Watchdog can be enabled by resetting the WDGEN bit.

9.1.3.2 Starting the Watchdog

In Watchdog mode the Timer is clocked by INTCLK/4.

If the Watchdog is software enabled, the time base must be written in the timer registers before entering Watchdog mode by resetting the WDGEN bit. Once reset, this bit cannot be changed by software.

If the Watchdog is hardware enabled, the time base is fixed by the reset value of the registers.

Resetting WDGEN causes the counter to start, regardless of the value of the Start-Stop bit.

In Watchdog mode, only the Prescaler Constant may be modified.

If the End of Count condition is reached a System Reset is generated.

TIMER/WATCHDOG (Cont'd)

9.1.3.3 Preventing Watchdog System Reset

In order to prevent a system reset, the sequence AAh, 55h must be written to WDTLR (Watchdog Timer Low Register). Once 55h has been written, the Timer reloads the constant and counting restarts from the preset value.

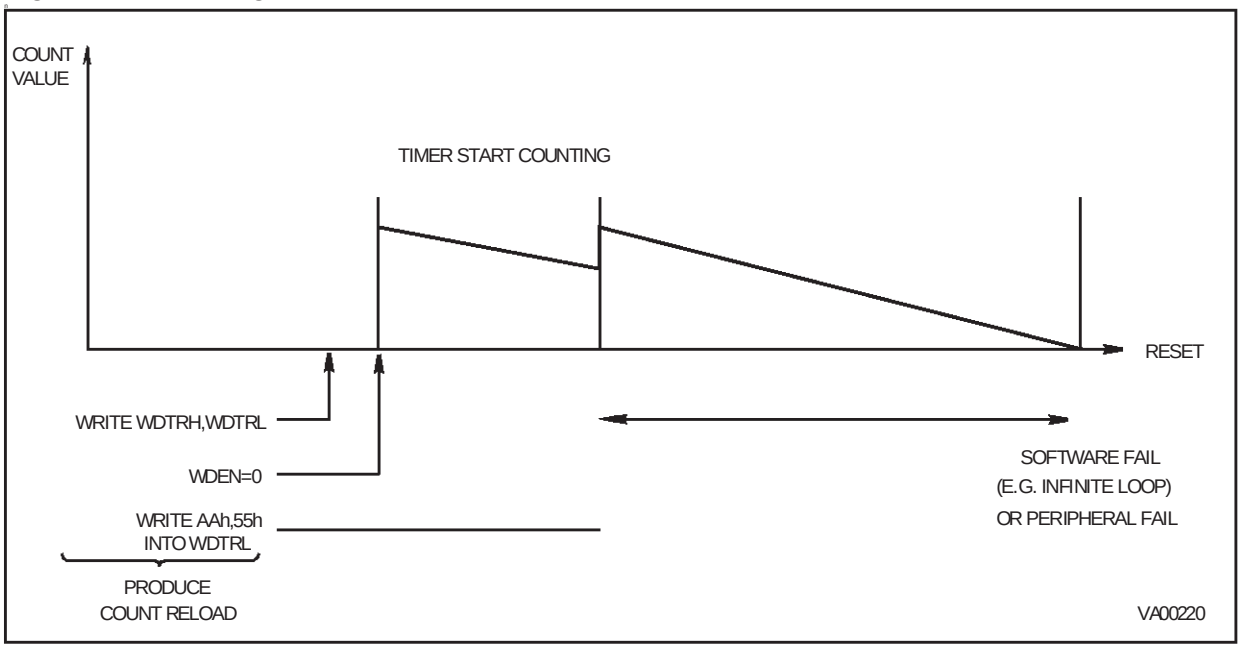
To reload the counter, the two writing operations must be performed sequentially without inserting other instructions that modify the value of the WDTLR register between the writing operations. The maximum allowed time between two reloads of the counter depends on the Watchdog timeout period.

9.1.3.4 Non-Stop Operation

In Watchdog Mode, a `Halt` instruction is regarded as illegal. Execution of the `Halt` instruction stops further execution by the CPU and interrupt acknowledgment, but does not stop INTCLK, CPUCLK or the Watchdog Timer, which will cause a System Reset when the End of Count condition is reached. Furthermore, ST_SP, S_C and the Input Mode selection bits are ignored. Hence, regardless of their status, the counter always runs in Continuous Mode, driven by the internal clock.

The Output mode should not be enabled, since in this context it is meaningless.

Figure 54. Watchdog Timer Mode



TIMER/WATCHDOG (Cont'd)

9.1.4 WDT Interrupts

The Timer/Watchdog issues an interrupt request at every End of Count, when this feature is enabled.

A pair of control bits, IA0S (EIVR.1, Interrupt A0 selection bit) and TLIS (EIVR.2, Top Level Input Selection bit) allow the selection of 2 interrupt sources (Timer/Watchdog End of Count, or External Pin) handled in two different ways, as a Top Level Non Maskable Interrupt (Software Reset), or as a source for channel A0 of the external interrupt logic.

A block diagram of the interrupt logic is given in Figure 55.

Note: Software traps can be generated by setting the appropriate interrupt pending bit.

Table 21 Interrupt Configuration below, shows all the possible configurations of interrupt/reset sources which relate to the Timer/Watchdog.

A reset caused by the watchdog will set bit 6, WDGRES of R242 (Clock Flag Register). See section CLOCK CONTROL REGISTERS.

Figure 55. Interrupt Sources

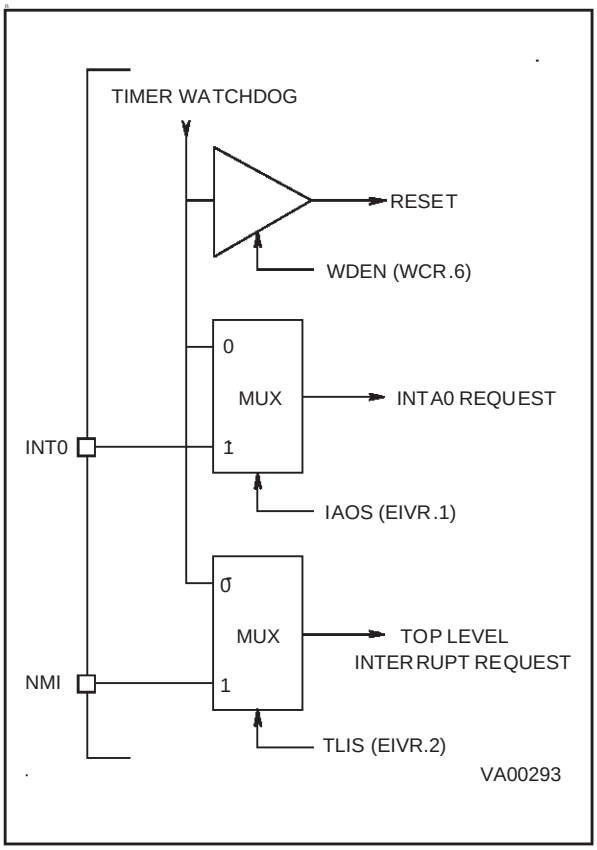


Table 21. Interrupt Configuration

Control Bits			Enabled Sources			Operating Mode
WDGEN	IAOS	TLIS	Reset	INTA0	Top Level	
0	0	0	WDG/Ext Reset	SW TRAP	SW TRAP	Watchdog
0	0	1	WDG/Ext Reset	SW TRAP	Ext Pin	Watchdog
0	1	0	WDG/Ext Reset	Ext Pin	SW TRAP	Watchdog
0	1	1	WDG/Ext Reset	Ext Pin	Ext Pin	Watchdog
1	0	0	Ext Reset	Timer	Timer	Timer
1	0	1	Ext Reset	Timer	Ext Pin	Timer
1	1	0	Ext Reset	Ext Pin	Timer	Timer
1	1	1	Ext Reset	Ext Pin	Ext Pin	Timer

Legend:
WDG = Watchdog function
SW TRAP = Software Trap

Note: If IAOS and TLIS = 0 (enabling the Watchdog EOC as interrupt source for both Top Level and INTA0 interrupts), only the INTA0 interrupt is taken into account.

TIMER/WATCHDOG (Cont'd)

9.1.5 Register Description

The Timer/Watchdog is associated with 4 registers mapped into Group F, Page 0 of the Register File.

WDTHR: Timer/Watchdog High Register

WDTLR: Timer/Watchdog Low Register

WDTPR: Timer/Watchdog Prescaler Register

WDTCR: Timer/Watchdog Control Register

Three additional control bits are mapped in the following registers on Page 0:

Watchdog Mode Enable, (WCR.6)

Top Level Interrupt Selection, (EIVR.2)

Interrupt A0 Channel Selection, (EIVR.1)

Note: The registers containing these bits also contain other functions. Only the bits relevant to the operation of the Timer/Watchdog are shown here.

Counter Register

This 16 bit register (WDTLR, WDTHR) is used to load the 16 bit counter value. The registers can be read or written "on the fly".

TIMER/WATCHDOG HIGH REGISTER(WDTHR)
R248 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

Bit 7:0 = **R[15:8]** *Counter Most Significant Bits*

TIMER/WATCHDOG LOW REGISTER(WDTLR)
R249 - Read/Write
Register Page: 0
Reset value: 1111 1111b (FFh)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

Bit 7:0 = **R[7:0]** *Counter Least Significant Bits.*

TIMER/WATCHDOG PRESCALER REGISTER (WDTPR)
R250 - Read/Write
Register Page: 0
Reset value: 1111 1111 (FFh)

7							0
PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0

Bit 7:0 = **PR[7:0]** *Prescaler value.*
A programmable value from 1 (00h) to 256 (FFh).

Warning: *In order to prevent incorrect operation of the Timer/Watchdog, the prescaler (WDTPR) and counter (WDTRL, WDTRH) registers must be initialised before starting the Timer/Watchdog. If this is not done, counting will start with the reset (un-initialised) values.*

WATCHDOG TIMER CONTROL REGISTER (WDTCR)
R251- Read/Write
Register Page: 0
Reset value: 0001 0010 (12h)

7							0
ST_SP	S_C	INMD1	INMD2	INEN	OUTMD	WROUT	OUTEN

Bit 7 = **ST_SP**: *Start/Stop Bit.*
This bit is set and cleared by software.
0: Stop counting
1: Start counting (see Warning above)

Bit 6 = **S_C**: *Single/Continuous.*
This bit is set and cleared by software.
0: Continuous Mode
1: Single Mode

Bit 5:4 = **INMD[1:2]**: *Input mode selection bits.*
These bits select the input mode:

INMD1	INMD2	INPUT MODE
0	0	Event Counter
0	1	Gated Input (Reset value)
1	0	Triggerable Input
1	1	Retriggerable Input

TIMER/WATCHDOG (Cont'd)

Bit 3 = **INEN**: *Input Enable.*
This bit is set and cleared by software.
0: Disable input section
1: Enable input section

Bit 2 = **OUTMD**: *Output Mode.*
This bit is set and cleared by software.
0: The output is toggled at every End of Count
1: The value of the WROUT bit is transferred to the output pin on every End Of Count if OUTEN=1.

Bit 1 = **WROUT**: *Write Out.*
The status of this bit is transferred to the Output pin when OUTMD is set; it is user definable to allow PWM output (on Reset WROUT is set).

Bit 0 = **OUTEN**: *Output Enable bit*
This bit is set and cleared by software.
0: Disable output
1: Enable output

WAIT CONTROL REGISTER (WCR)
R252 - Read/Write
Register Page: 0
Reset value: 0111 1111 (7Fh)

7							0
x	WDGEN	x	x	x	x	x	x

Bit 6 = **WDGEN**: *Watchdog Enable* (active low).
Resetting this bit via software enters the Watchdog mode. Once reset, it cannot be set anymore by the user program. At System Reset, the Watchdog mode is disabled.

Note: This bit is ignored if the Hardware Watchdog option is enabled by pin HW0SW1 (if available).

EXTERNAL INTERRUPT VECTOR REGISTER (EIVR)
R246 - Read/Write
Register Page: 0
Reset value: xxxx 0110 (x6h)

7							0
x	x	x	x	x	TLIS	IA0S	x

Bit 2 = **TLIS**: *Top Level Input Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is TL interrupt source
1: NMI is TL interrupt source

Bit 1 = **IA0S**: *Interrupt Channel A0 Selection.*
This bit is set and cleared by software.
0: Watchdog End of Count is INTA0 source
1: External Interrupt pin is INTA0 source

Warning: *To avoid spurious interrupt requests, the IA0S bit should be accessed only when the interrupt logic is disabled (i.e. after the DI instruction). It is also necessary to clear any possible interrupt pending requests on channel A0 before enabling this interrupt channel. A delay instruction (e.g. a NOP instruction) must be inserted between the reset of the interrupt pending bit and the IA0S write instruction.*

Other bits are described in the Interrupt section.

9.2 MULTIFUNCTION TIMER (MFT)

9.2.1 Introduction

The Multifunction Timer (MFT) peripheral offers powerful timing capabilities and features 12 operating modes, including automatic PWM generation and frequency measurement.

The MFT comprises a 16-bit Up/Down counter driven by an 8-bit programmable prescaler. The input clock may be INTCLK/3 or an external source. The timer features two 16-bit Comparison Registers, and two 16-bit Capture/Load/Reload Registers. Two input pins and two alternate function output pins are available.

Several functional configurations are possible, for instance:

– 2 input captures on separate external lines, and 2 independent output compare functions with the counter in free-running mode, or 1 output compare at a fixed repetition rate.

– 1 input capture, 1 counter reload and 2 independent output compares.

– 2 alternate autoreloads and 2 independent output compares.

– 2 alternate captures on the same external line and 2 independent output compares at a fixed repetition rate.

When two timers are present in an ST9 device, a combined operating mode is available.

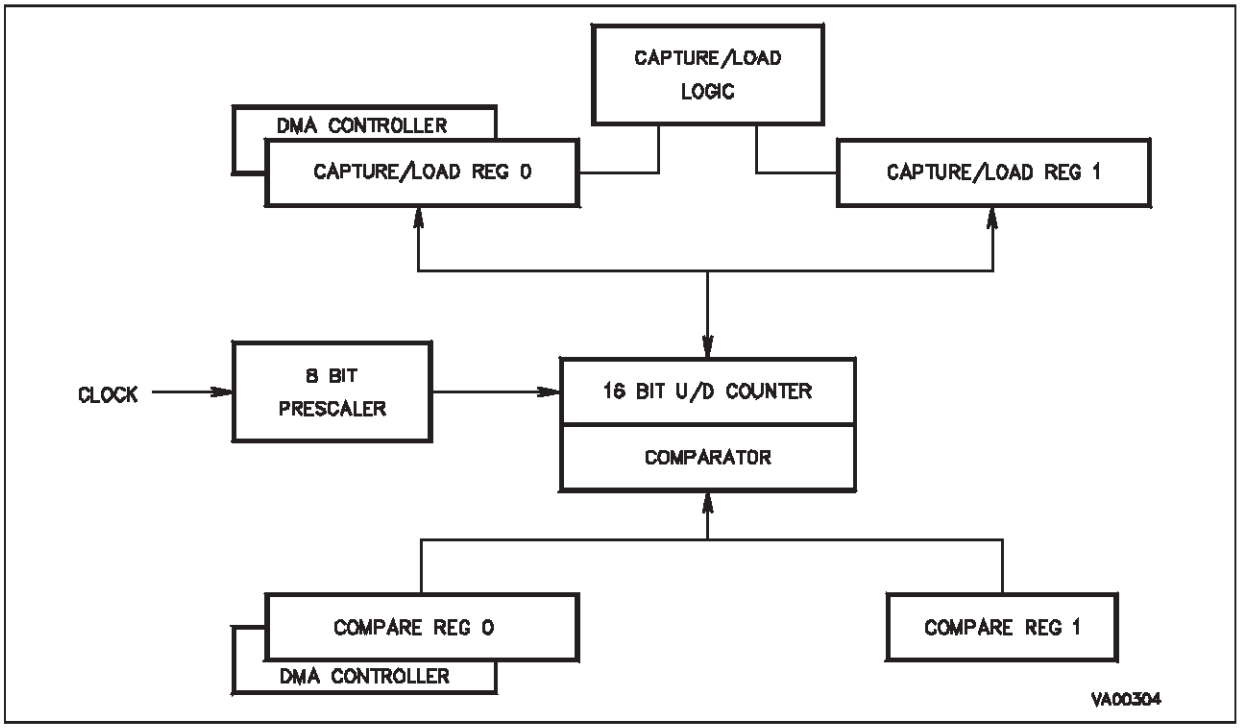
Four internal signals are also available for timing on-chip functions: the On-Chip Event signal can be used to control other peripherals on the chip itself, and 3 other signals, can be internally connected to I/O ports, to allow automatic, timed, DMA transfers. The two external inputs may be individually programmed to detect any of the following:

– rising edges

– falling edges

– both rising and falling edges

Figure 56. MFT Simplified Block Diagram



MULTIFUNCTION TIMER (Cont'd)

The configuration of each input is programmed in the Input Control Register.

Each of the two output pins can be driven from any of three possible sources:

– Compare Register 0 logic

– Compare Register 1 logic

– Overflow/Underflow logic

Each of these three sources can cause one of the following four actions, independently, on each of the two outputs:

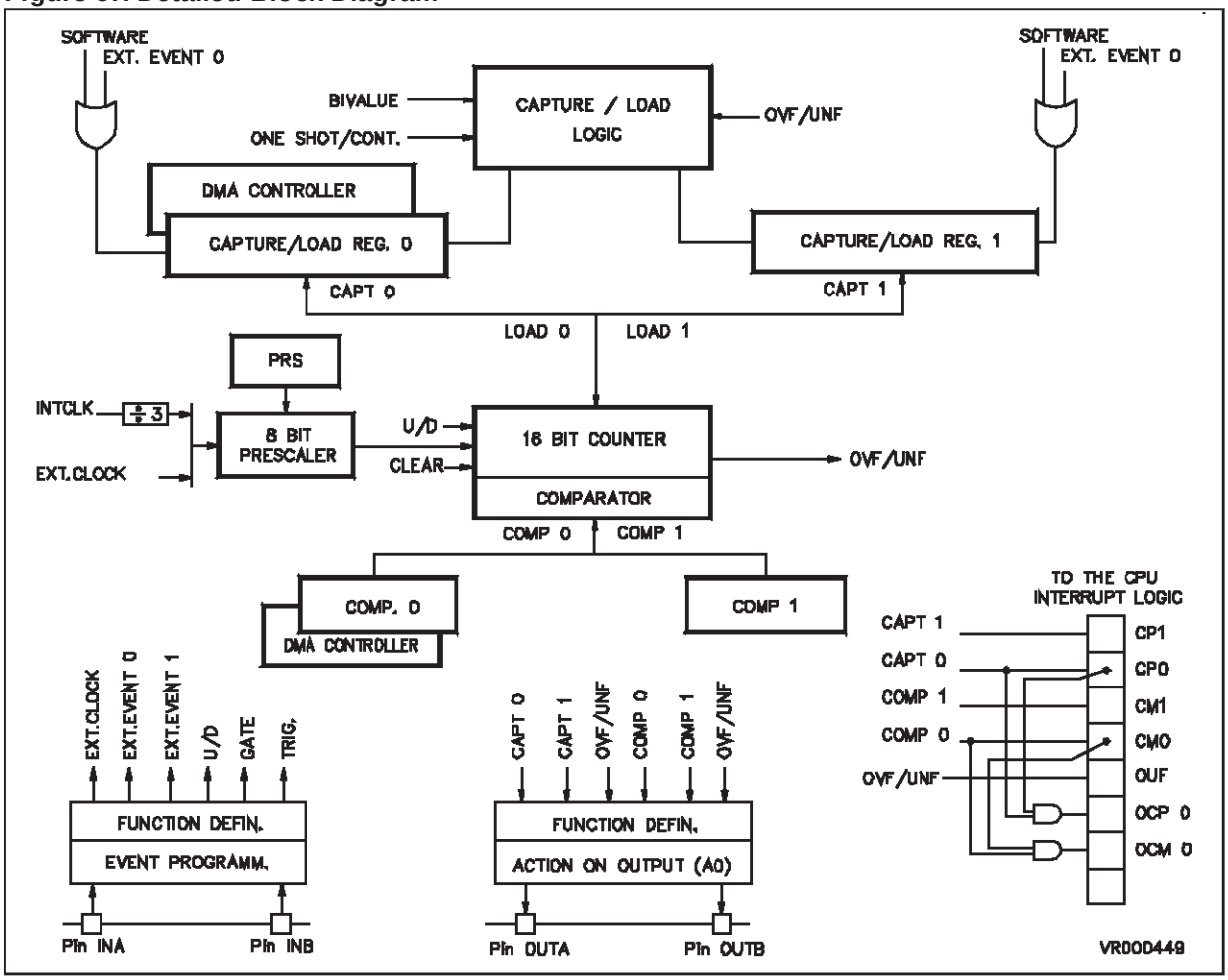
– Nop, Set, Reset, Toggle

In addition, an additional On-Chip Event signal can be generated by two of the three sources mentioned above, i.e. Over/Underflow event and Compare 0 event. This signal can be used internally to synchronise another on-chip peripheral. Five maskable interrupt sources referring to an End Of Count condition, 2 input captures and 2 output compares, can generate 3 different interrupt requests (with hardware fixed priority), pointing to 3 interrupt routine vectors.

Two independent DMA channels are available for rapid data transfer operations. Each DMA request (associated with a capture on the REG0R register, or with a compare on the CMP0R register) has priority over an interrupt request generated by the same source.

A SWAP mode is also available to allow high speed continuous transfers (see Interrupt and DMA chapter).

Figure 57. Detailed Block Diagram



MULTIFUNCTION TIMER (Cont'd)

9.2.2 Functional Description

The MFT operating modes are selected by programming the Timer Control Register (TCR) and the Timer Mode Register (TMR).

9.2.2.1 One Shot Mode

When the counter generates an overflow (in up-count mode), or an underflow (in down-count mode), that is to say when an End Of Count condition is reached, the counter stops and no counter reload occurs. The counter may only be restarted by an external or software trigger. The One Shot Mode is entered by setting the CO bit in TMR.

9.2.2.2 Continuous Mode

Whenever the counter reaches an End Of Count condition, the counting sequence is automatically restarted and the counter is reloaded from REG0R (or from REG1R, when selected in Biload Mode). Continuous Mode is entered by resetting the C0 bit in TMR.

9.2.2.3 Triggered And Retriggered Modes

A triggered event may be generated by software (by setting either the CP0 or the CP1 bit in the FLAGR timer register), or by an external source which may be programmed to respond to the rising edge, the falling edge or both, by programming bits A0-A1 and B0-B1 in ICR.

In One Shot and Triggered Mode, every trigger event arriving before an End Of Count, is masked. In One Shot and Retriggered Mode, every trigger received while the counter is running, automatically reloads the counter from REG0R. Triggered/Retriggered Mode is set by the REN bit in TMR.

The TxINA input refers to REG0R and the TxINB input refers to REG1R.

WARNING. If the Triggered Mode is selected when the counter is in Continuous Mode, every trigger is disabled, it is not therefore possible to synchronise the counting cycle by hardware or software.

9.2.2.4 Gated Mode

In this mode, counting takes place only when the external gate input is at a logic low level. The selection of TxINA or TxINB as the gate input is made by programming the IN0-IN3 bits in ICR.

9.2.2.5 Capture Mode

The REG0R and REG1R registers may be independently set in Capture Mode by setting RM0 or RM1 in TMR, so that a capture of the current count value can be performed either on REG0R or on REG1R, initiated by software (by setting CP0 or CP1 in the FLAGR register) or by an event on the external input pins.

WARNING. Care should be taken when two software captures are to be performed on the same register. In this case, at least one instruction must be present between the first CP0/CP1 bit set and the subsequent CP0/CP1 bit reset instructions.

9.2.2.6 Up/Down Mode

The counter can count up or down depending on the state of the UDC bit (Up/Down Count) in TCR, or on the configuration of the external input pins, which have priority over UDC (see Input pin assignment in ICR). The UDCS bit returns the counter up/down current status (see also the Up/Down Autodiscrimination mode in the Input Pin Assignment Section).

MULTIFUNCTION TIMER (Cont'd)

9.2.2.7 Free Running Mode

The timer counts continuously (in up or down mode) and the counter value simply overflows or underflows through FFFFh or zero; there is no End Of Count condition as such, and no reloading takes place. This mode is automatically selected either in Bicapture Mode or by setting REG0R for a capture function (Continuous Mode must also be set). In Autoclear Mode, free running operation can be had, with the possibility of choosing a maximum count value before overflow or underflow which is less than 2^{16} (see Autoclear Mode).

9.2.2.8 Monitor Mode

When the RM1 bit in TMR is reset, and the timer is not in Bivalue Mode, REG1R acts as a monitor, duplicating the current up or down counter contents, thus allowing the counter to be read "on the fly".

9.2.2.9 Autoclear Mode

A clear command forces the counter either to 0000h or to FFFFh, depending on whether upcounting or downcounting is selected. The counter reset may be obtained either directly, through the CCL bit in TCR, or by entering the Autoclear Mode, through the CCP0 and CCMP0 bits in TCR.

Every capture performed on REG0R (if CCP0 is set), or every successful compare performed by CMP0R (if CCMP0 is set), clears the counter and reloads the prescaler.

The Clear On Capture mode allows direct measurement of delta time between successive captures on REG0R, while the Clear On Compare mode allows free running with the possibility of choosing a maximum count value before overflow or underflow which is less than 2^{16} (see Free Running Mode).

9.2.2.10 Bivalue Mode

Depending on the value of the RM0 bit in TMR, the Biload Mode (RM0 reset) or the Bicapture Mode (RM0 set) can be selected as illustrated inFigure 22 below:

Table 22. Bivalue Modes

TMR bits			Timer Operating Modes
RM0	RM1	BM	
0	X	1	BiLoad mode
1	X	1	BiCapture Mode

A) Biload Mode

The Biload Mode is entered by selecting the Bivalue Mode (BM set in TMR) and programming REG0R as a reload register (RM0 reset in TMR).

At any End Of Count, counter reloading is performed alternately from REG0R and REG1R, (a low level for BM bit always sets REG0R as the current register, so that, after a Low to High transition of BM bit, the first reload is always from REG0R).

MULTIFUNCTION TIMER (Cont'd)

Every software or external trigger event on REG0R performs a reload from REG0R resetting the Biload cycle. In One Shot mode (reload initiated by software or by an external trigger), reloading is always from REG0R.

B) Bicapture Mode

The Bicapture Mode is entered by selecting the Bivalue Mode (the BM bit in TMR is set) and by programming REG0R as a capture register (the RM0 bit in TMR is set).

Every capture event, software simulated (by setting the CP0 flag) or coming directly from the TxINA input line, captures the current counter value alternately into REG0R and REG1R. When the BM bit is reset, REG0R is the current register, so that the first capture, after resetting the BM bit, is always into REG0R.

9.2.2.11 Parallel Mode

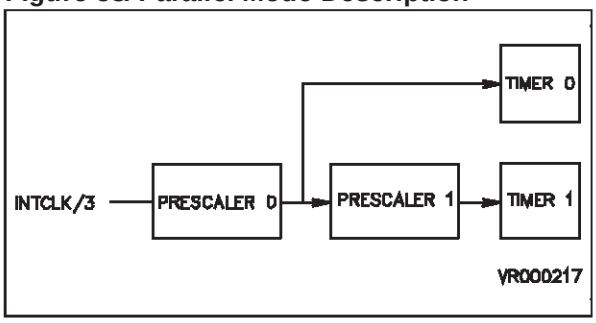
When two timers are present on an ST9 device, the parallel mode is entered when the ECK bit in the TMR register of Timer 1 is set. The Timer 1 prescaler input is internally connected to the Timer 0 prescaler output. Timer 0 prescaler input is connected to the system clock line.

By loading the Prescaler Register of Timer 1 with the value 00h the two timers (Timer 0 and Timer 1) are driven by the same frequency in parallel mode. In this mode the clock frequency may be divided by a factor in the range from 1 to 2^{16}.

9.2.2.12 Autodiscriminator Mode

The phase difference sign of two overlapping pulses (respectively on TxINB and TxINA) generates a one step up/down count, so that the up/down control and the counter clock are both external. The setting of the UDC bit in the TCR register has no effect in this configuration.

Figure 58. Parallel Mode Description



MULTIFUNCTION TIMER (Cont'd)

9.2.3 Input Pin Assignment

The two external inputs (TxINA and TxINB) of the timer can be individually configured to catch a particular external event (i.e. rising edge, falling edge, or both rising and falling edges) by programming the two relevant bits (A0, A1 and B0, B1) for each input in the external Input Control Register (ICR).

The 16 different functional modes of the two external inputs can be selected by programming bits IN0 - IN3 of the ICR, as illustrated inFigure 23

Table 23. Input Pin Function

I C Reg. IN3-IN0 bits	TxINA Input Function	TxINB Input Function
0000	not used	not used
0001	not used	Trigger
0010	Gate	not used
0011	Gate	Trigger
0100	not used	Ext. Clock
0101	Trigger	not used
0110	Gate	Ext. Clock
0111	Trigger	Trigger
1000	Clock Up	Clock Down
1001	Up/Down	Ext. Clock
1010	Trigger Up	Trigger Down
1011	Up/Down	not used
1100	Autodiscr.	Autodiscr.
1101	Trigger	Ext. Clock
1110	Ext. Clock	Trigger
1111	Trigger	Gate

Some choices relating to the external input pin assignment are defined in conjunction with the RM0 and RM1 bits in TMR.

For input pin assignment codes which use the input pins as Trigger Inputs (except for code 1010, Trigger Up:Trigger Down), the following conditions apply:

– a trigger signal on the TxINA input pin performs an U/D counter load if RM0 is reset, or an external capture if RM0 is set.

– a trigger signal on the TxINB input pin always performs an external capture on REG1R. The TxINB input pin is disabled when the Bivalue Mode is set.

Note: For proper operation of the External Input pins, the following must be observed:

– the minimum external clock/trigger pulse width must not be less than the system clock (INTCLK) period if the input pin is programmed as rising or falling edge sensitive.

– the minimum external clock/trigger pulse width must not be less than the prescaler clock period (INTCLK/3) if the input pin is programmed as rising and falling edge sensitive (valid also in Auto discrimination mode).

– the minimum delay between two clock/trigger pulse active edges must be greater than the prescaler clock period (INTCLK/3), while the minimum delay between two consecutive clock/trigger pulses must be greater than the system clock (INTCLK) period.

– the minimum gate pulse width must be at least twice the prescaler clock period (INTCLK/3).

– in Autodiscrimination mode, the minimum delay between the input pin A pulse edge and the edge of the input pin B pulse, must be at least equal to the system clock (INTCLK) period.

– if a number, N, of external pulses must be counted using a Compare Register in External Clock mode, then the Compare Register must be loaded with the value [X +/- (N-1)], where X is the starting counter value and the sign is chosen depending on whether Up or Down count mode is selected.

MULTIFUNCTION TIMER (Cont'd)

9.2.3.1 TxINA = I/O - TxINB = I/O

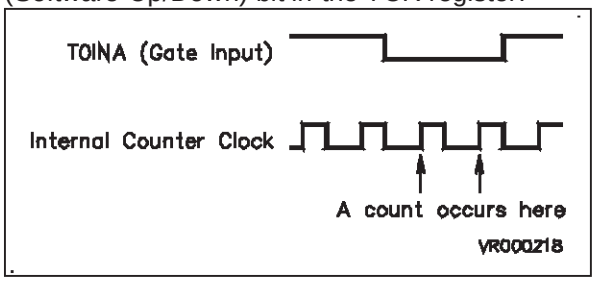
Input pins A and B are not used by the Timer. The counter clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

9.2.3.2 TxINA = I/O - TxINB = Trigger

The signal applied to input pin B acts as a trigger signal on REG1R register. The prescaler clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

9.2.3.3 TxINA = Gate - TxINB = I/O

The signal applied to input pin A acts as a gate signal for the internal clock (i.e. the counter runs only when the gate signal is at a low level). The counter clock is internally generated and the up/down control may be made only by software via the UDC (Software Up/Down) bit in the TCR register.



9.2.3.4 TxINA = Gate - TxINB = Trigger

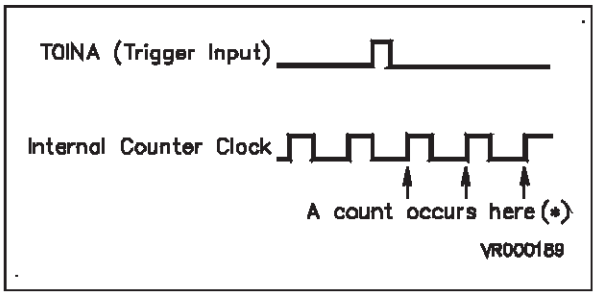
Both input pins A and B are connected to the timer, with the resulting effect of combining the actions relating to the previously described configurations.

9.2.3.5 TxINA = I/O - TxINB = Ext. Clock

The signal applied to input pin B is used as the external clock for the prescaler. The up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

9.2.3.6 TxINA = Trigger - TxINB = I/O

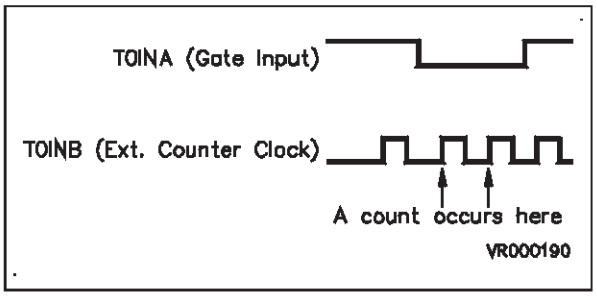
The signal applied to input pin A acts as a trigger for REG0R, initiating the action for which the register was programmed (i.e. a reload or capture). The prescaler clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.



(*) The timer is in One shot mode and REGOR in Reload mode

9.2.3.7 TxINA = Gate - TxINB = Ext. Clock

The signal applied to input pin B, gated by the signal applied to input pin A, acts as external clock for the prescaler. The up/down control may be made only by software action through the UDC bit in the TCR register.



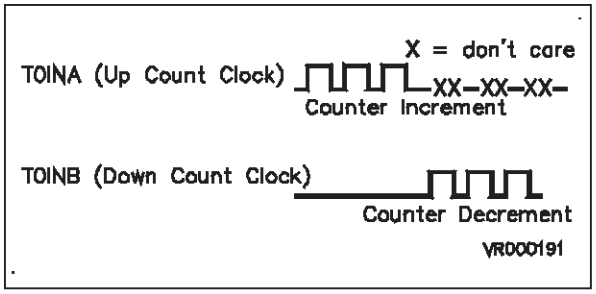
9.2.3.8 TxINA = Trigger - TxINB = Trigger

The signal applied to input pin A (or B) acts as trigger signal for REG0R (or REG1R), initiating the action for which the register has been programmed. The counter clock is internally generated and the up/down selection may be made only by software via the UDC (Software Up/Down) bit in the TCR register.

MULTIFUNCTION TIMER (Cont'd)

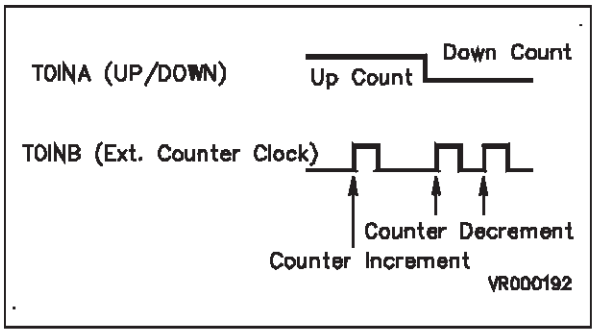
9.2.3.9 TxINA = Clock Up - TxINB = Clock Down

The edge received on input pin A (or B) performs a one step up (or down) count, so that the counter clock and the up/down control are external. Setting the UDC bit in the TCR register has no effect in this configuration, and input pin B has priority on input pin A.



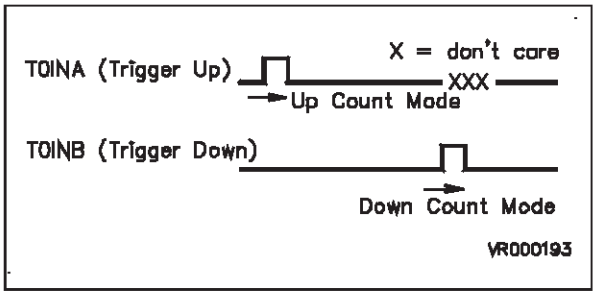
9.2.3.10 TxINA = Up/Down - TxINB = Ext Clock

An High (or Low) level applied to input pin A sets the counter in the up (or down) count mode, while the signal applied to input pin B is used as clock for the prescaler. Setting the UDC bit in the TCR register has no effect in this configuration.



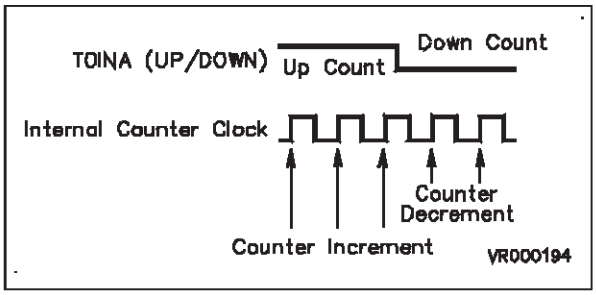
9.2.3.11 TxINA = Trigger Up - TxINB = Trigger Down

Up/down control is performed through both input pins A and B. A edge on input pin A sets the up count mode, while a edge on input pin B (which has priority on input pin A) sets the down count mode. The counter clock is internally generated, and setting the UDC bit in the TCR register has no effect in this configuration.



9.2.3.12 TxINA = Up/Down - TxINB = I/O

An High (or Low) level of the signal applied on input pin A sets the counter in the up (or down) count mode. The counter clock is internally generated. Setting the UDC bit in the TCR register has no effect in this configuration.

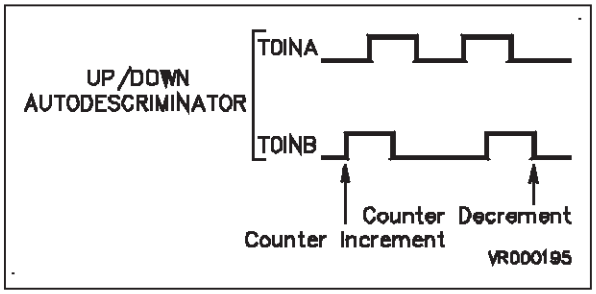


MULTIFUNCTION TIMER (Cont'd)

9.2.3.13 Autodiscrimination Mode

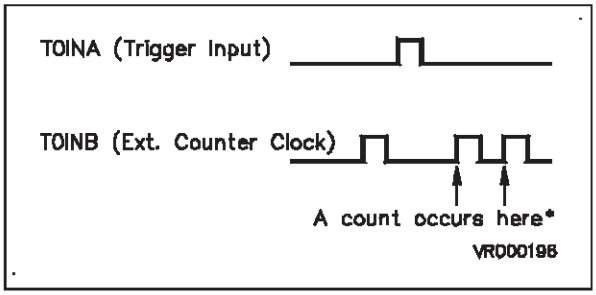
The phase between two pulses (respectively on input pin B and input pin A) generates a one step up (or down) count, so that the up/down control and the counter clock are both external. Thus, if the rising edge of TxINB arrives when TxINA is at a low level, the timer is incremented (no action if the rising edge of TxINB arrives when TxINA is at a high level). If the falling edge of TxINB arrives when TxINA is at a low level, the timer is decremented (no action if the falling edge of TxINB arrives when TxINA is at a high level).

Setting the UDC bit in the TCR register has no effect in this configuration.



9.2.3.14 TxINA = Trigger - TxINB = Ext. Clock

The signal applied to input pin A acts as a trigger signal on REG0R, initiating the action for which the register was programmed (i.e. a reload or capture), while the signal applied to input pin B is used as the clock for the prescaler.



(*) The timer is in One shot mode and REG0R in reload mode

9.2.3.15 TxINA = Ext. Clock - TxINB = Trigger

The signal applied to input pin B acts as a trigger, performing a capture on REG1R, while the signal applied to input pin A is used as the clock for the prescaler.

9.2.3.16 TxINA = Trigger - TxINB = Gate

The signal applied to input pin A acts as a trigger signal on REG0R, initiating the action for which the register was programmed (i.e. a reload or capture), while the signal applied to input pin B acts as a gate signal for the internal clock (i.e. the counter runs only when the gate signal is at a low level).

MULTIFUNCTION TIMER (Cont'd)

9.2.4 Output Pin Assignment

Two external outputs are available for each timer when programmed as Alternate Function Outputs of the I/O pins.

Two registers for every timer, Output A Control Register (OACR) and Output B Control Register (OBCR) define the driver for the outputs and the actions to be performed.

Each of the two output pins can be driven from any of the three possible sources:

– Compare Register 0 event logic

– Compare Register 1 event logic

– Overflow/Underflow event logic.

Each of these three sources can cause one of the following four actions on any of the two outputs:

– Nop

– Set

– Reset

– Toggle

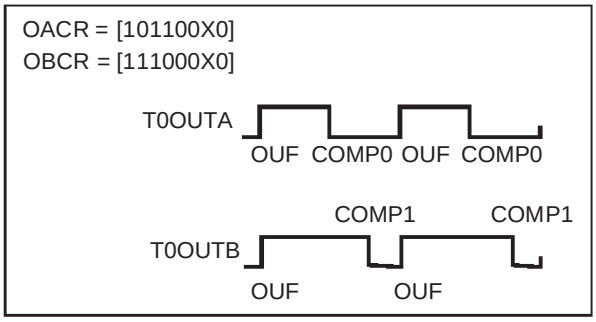
Furthermore an On Chip Event signal can be driven by two of the three sources: the Over/Underflow event and Compare 0 event by programming the CEV bit of the OACR register and the OEV bit of OBCR register respectively. This signal can be used internally to synchronise another on-chip peripheral.

Output Waveforms

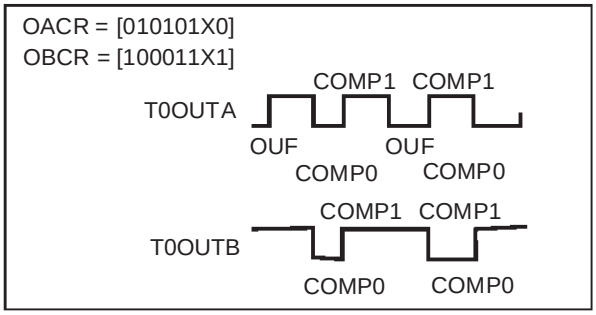
Depending on the programming of OACR and OBCR, the following example waveforms can be generated on TxOUTA and TxOUTB pins.

For a configuration where TxOUTA is driven by the Over/Underflow (OUF) and the Compare 0 event (CM0), and TxOUTB is driven by the Over/Underflow and Compare 1 event (CM1):

OACR is programmed with TxOUTA preset to "0", OUF sets TxOUTA, CM0 resets TxOUTA and CM1 does not affect the output.
OBCR is programmed with TxOUTB preset to "0", OUF sets TxOUTB, CM1 resets TxOUTB while CM0 does not affect the output.

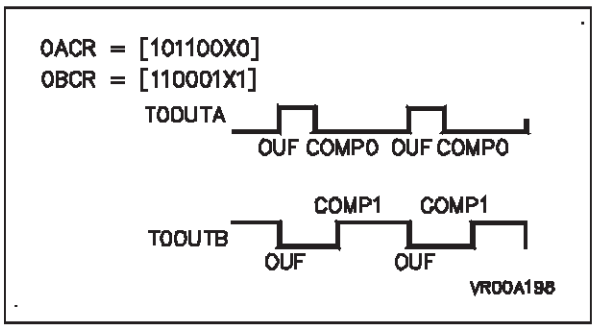


For a configuration where TxOUTA is driven by the Over/Underflow, by Compare 0 and by Compare 1; TxOUTB is driven by both Compare 0 and Compare 1. OACR is programmed with TxOUTA preset to "0". OUF toggles Output 0, as do CM0 and CM1. OBCR is programmed with TxOUTB preset to "1". OUF does not affect the output; CM0 resets TxOUTB and CM1 sets it.

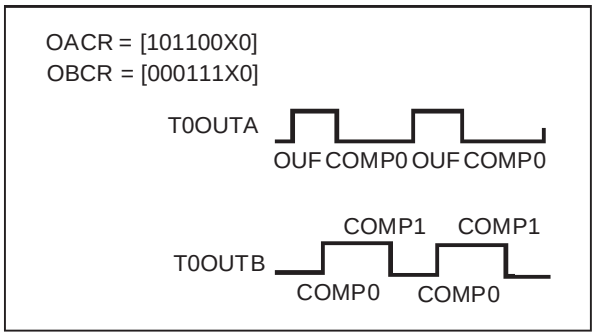


MULTIFUNCTION TIMER (Cont'd)

For a configuration where TxOUTA is driven by the Over/Underflow and by Compare 0, and TxOUTB is driven by the Over/Underflow and by Compare 1. OACR is programmed with TxOUTA preset to "0". OUF sets TxOUTA while CM0 resets it, and CM1 has no effect. OBCR is programmed with TxOUTB preset to "1". OUF toggles TxOUTB, CM1 sets it and CM0 has no effect.



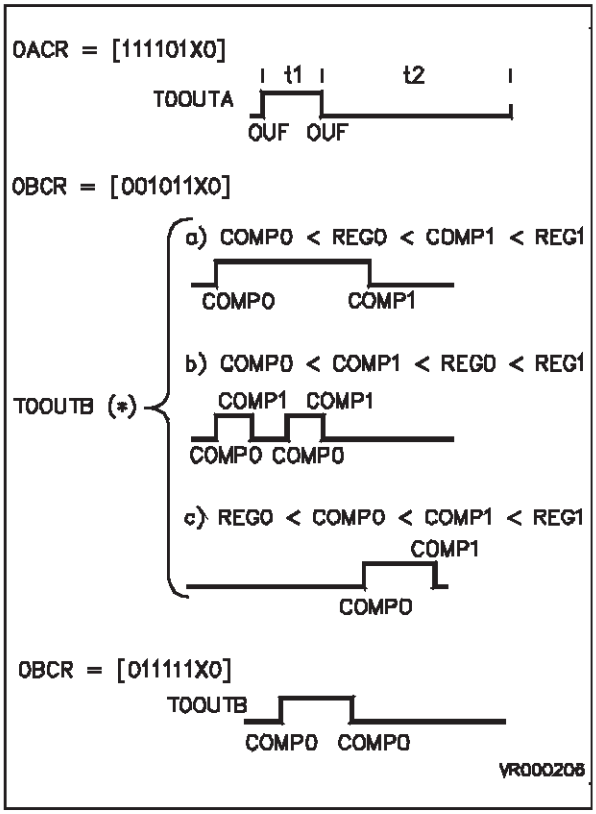
For a configuration where TxOUTA is driven by the Over/Underflow and by Compare 0, and TxOUTB is driven by Compare 0 and 1. OACR is programmed with TxOUTA preset to "0". OUF sets TxOUTA, CM0 resets it and CM1 has no effect. OBCR is programmed with TxOUTB preset to "0". OUF has no effect, CM0 sets TxOUTB and CM1 toggles it.



Output Waveform Samples In Biload Mode

TxOUTA is programmed to monitor the two time intervals, t1 and t2, of the Biload Mode, while TxOUTB is independent of the Over/Underflow and is driven by the different values of Compare 0 and Compare 1. OACR is programmed with TxOUTA preset to "0". OUF toggles the output and CM0 and CM1 do not affect TxOUTA. OBCR is programmed with TxOUTB preset to "0". OUF has no effect, while CM1 resets TxOUTB and CM0 sets it.

Depending on the CM1/CM0 values, three different sample waveforms have been drawn based on the above mentioned configuration of OBCR. In the last case, with a different programmed value of OBCR, only Compare 0 drives TxOUTB, toggling the output.



Note (*) Depending on the CMP1R/CMP0R values

MULTIFUNCTION TIMER (Cont'd)

9.2.5 Interrupt and DMA

9.2.5.1 Timer Interrupt

The timer has 5 different Interrupt sources, belonging to 3 independent groups, which are assigned to the following Interrupt vectors:

Table 24. Timer Interrupt Structure

Interrupt Source	Vector Address
COMP 0 COMP 1	xxxx x110
CAPT 0 CAPT 1	xxxx x100
Overflow/Underflow	xxxx x000

The three least significant bits of the vector pointer address represent the relative priority assigned to each group, where 000 represents the highest priority level. These relative priorities are fixed by hardware, according to the source which generates the interrupt request. The 5 most significant bits represent the general priority and are programmed by the user in the Interrupt Vector Register (IVR) associated with each Timer.

Each source can be masked by a dedicated bit in the Interrupt/DMA Mask Register (IDMR) of each timer, as well as by a global mask enable bit (IDMR.7) which masks all interrupts.

If an interrupt request (CM0 or CP0) is present before the corresponding pending bit is reset, an overrun condition occurs. This condition is flagged in two dedicated overrun bits, relating to the Comp0 and Capt0 sources, in the Timer Flag Register (FLAGR).

9.2.5.2 Timer DMA

Two Independent DMA channels, associated with Comp0 and Capt0 respectively, allow DMA transfers from Register File or Memory to the Comp0 Register, and from the Capt0 Register to Register File or Memory). Their priority is set by hardware as follows:

– Compare 0 Destination — Lower Priority

– Capture 0 Source — Higher Priority

The two DMA request sources are independently maskable by two DMA Mask bits, mapped in the Timer Interrupt/DMA Mask register (IDMR).

The two End of Block procedures, associated with each Interrupt mask and DMA mask combination, follow the standard architecture described in the Interrupt and DMA chapters.

9.2.5.3 DMA Pointers

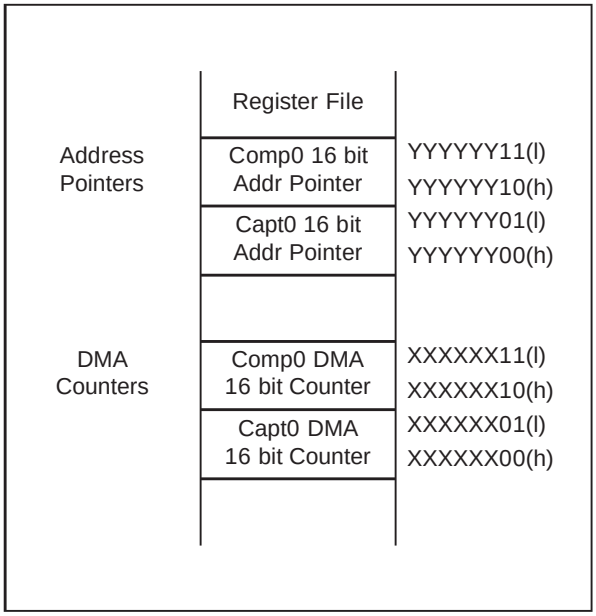
The 6 programmable most significant bits of the DMA Counter Pointer Register (DCPR) and of the DMA Address Pointer Register (DAPR) are common to both channels (Comp0 and Capt0). The Comp0 and Capt0 Address Pointers are mapped as a pair in the Register File, as are the Comp0 and Capt0 DMA Counter pair.

In order to specify either the Capt0 or the Comp0 pointers, according to the channel being serviced, the Timer resets address bit 1 for CAPT0 and sets it for COMP0, when the D0 bit in the DCPR register is equal to zero (Word address in Register File). In this case (transfers between peripheral registers and memory), the pointers are split into two groups of adjacent Address and Counter pairs respectively.

For peripheral register to register transfers (selected by programming "1" into bit 0 of the DCPR register), only one pair of pointers is required, and the pointers are mapped into one group of adjacent positions.

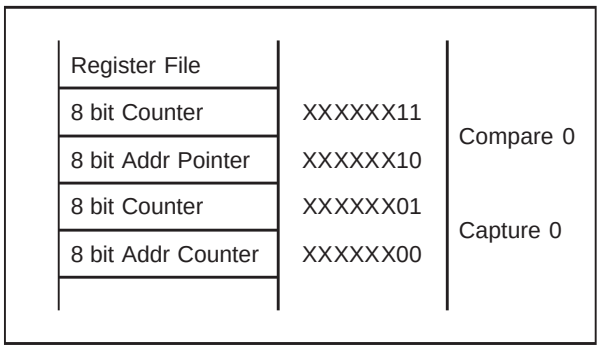
The DMA Address Pointer Register (DAPR) is not used in this case, but must be considered reserved.

Figure 59. Pointer Mapping for Transfers between Registers and Memory



MULTIFUNCTION TIMER (Cont'd)

Figure 60. Pointer Mapping for Register to Register Transfers



9.2.5.4 DMA Transaction Priorities

Each Timer DMA transaction is a 16-bit operation, therefore two bytes must be transferred sequentially, by means of two DMA transfers. In order to speed up each word transfer, the second byte transfer is executed by automatically forcing the peripheral priority to the highest level (000), regardless of the previously set level. It is then restored to its original value after executing the transfer. Thus, once a request is being serviced, its hardware priority is kept at the highest level regardless of the other Timer internal sources, i.e. once a Comp0 request is being serviced, it maintains a higher priority, even if a Capt0 request occurs between the two byte transfers.

9.2.5.5 DMA Swap Mode

After a complete data table transfer, the transaction counter is reset and an End Of Block (EOB) condition occurs, the block transfer is completed.

The End Of Block Interrupt routine must at this point reload both address and counter pointers of the channel referred to by the End Of Block interrupt source, if the application requires a continuous high speed data flow. This procedure causes speed limitations because of the time required for the reload routine.

The SWAP feature overcomes this drawback, allowing high speed continuous transfers. Bit 2 of the DMA Counter Pointer Register (DCPR) and of the DMA Address Pointer Register (DAPR), toggles after every End Of Block condition, alternately providing odd and even address (D2-D7) for the pair of pointers, thus pointing to an updated pair, after a block has been completely transferred. This allows the User to update or read the first block and to update the pointer values while the second is being transferred. These two toggle bits are software writable and readable, mapped in DCPR bit 2 for the CM0 channel, and in DAPR bit 2 for the CP0 channel (though a DMA event on a channel, in Swap mode, modifies a field in DAPR and DCPR common to both channels, the DAPR/DCPR content used in the transfer is always the bit related to the correct channel).

The SWAP mode can be enabled by a control bit placed in the Interrupt Control Register.

***WARNING**: this mode is always set for both channels (CM0 and CP0).*

9.2.5.6 DMA End Of Block Interrupt Routine

An interrupt request is generated after each block transfer (EOB) and its priority is the same as that assigned in the usual interrupt request, for the two channels. As a consequence, they will be serviced only when no DMA request occurs, and will be subject to a possible OUF Interrupt request, which has higher priority.

The following is a typical EOB procedure (with swap mode enabled):

– Test Toggle bit and Jump.
– Reload Pointers (odd or even depending on toggle bit status).
– Reset EOB bit: this bit must be reset only after the old pair of pointers has been restored, so that, if a new EOB condition occurs, the next pair of pointers is ready for swapping.
– Verify the software protection condition (see Section 9.2.5.7).
– Read the corresponding Overrun bit: this confirms that no DMA request has been lost in the meantime.
– Return.

WARNING: The EOB bits are read/write **only** for test purposes. Writing a logical "1" by software (when the SWEN bit is set) will cause a spurious interrupt request. These bits are normally only reset by software.

9.2.5.7 DMA Software Protection

A second EOB condition may occur before the first EOB routine is completed, this would cause a not yet updated pointer pair to be addressed, with consequent overwriting of memory. To prevent these errors, a protection mechanism is provided, such that the attempted setting of the EOB bit before it has been reset by software will cause the DMA mask on that channel to be reset (DMA disabled), thus blocking any further DMA operation. As shown above, this mask bit should always be checked in each EOB routine, to ensure that all DMA transfers are properly served.

MULTIFUNCTION TIMER (Cont'd)

9.2.6 Register Description

Twentyone control and data registers are associated with each Multifunction Timer, and are located in the Group F I/O pages of the ST9 Register File, as shown in Figure 25 below.

Note that unused registers must be considered as reserved.

Each register is described in detail in the following pages, together with the meaning and function of every bit.

Note: In the register description on the following pages, register and page numbers are given using the example of Timer 0.

Table 25. Multifunction Timer Register Map (Group F)

	Page 8	Page 9	Page 10 Page 0Ah	Page 12 Page 0Ch	Page 13 Page 0Dh	
R255	IDMR - TIM1		IDMR - TIM0	IDMR - TIM3		FFh
R254	FLAGR-TIM1		FLAGR - TIM0	FLAGR - TIM3		FEh
R253	OBCR - TIM1		OBCR - TIM0	OBCR - TIM3		FDh
R252	OACR - TIM1		OACR - TIM0	OACR - TIM3		FCh
R251	PRSR - TIM1		PRSR - TIM0	PRSR - TIM3		FBh
R250	ICR - TIM1		ICR - TIM0	ICR - TIM3		FAh
R249	TMR - TIM1		TMR - TIM0	TMR - TIM3		F9h
R248	TCR - TIM1	IOCR - TIM0, TIM1	TCR - TIM0	TCR - TIM3	IOCR - TIM3	F8h
R247	CMP1LR - TIM1	IDCR - TIM1	CMP1LR - TIM0	CMP1LR - TIM3	IDCR - TIM3	F7h
R246	CMP1HR - TIM1	IVR - TIM1	CMP1HR - TIM0	CMP1HR - TIM3	IVR - TIM3	F6h
R245	CMP0LR - TIM1	DAPR - TIM1	CMP0LR - TIM0	CMP0LR - TIM3	DAPR - TIM3	F5h
R244	CMP0HR - TIM1	DCPR - TIM1	CMP0HR - TIM0	CMP0HR - TIM3	DCPR - TIM3	F4h
R243	REG1LR - TIM1	IDCR - TIM0	REG1LR - TIM0	REG1LR - TIM3		F3h
R242	REG1HR - TIM1	IVR - TIM0	REG1HR - TIM0	REG1HR - TIM3		F2h
R241	REG0LR - TIM1	DAPR - TIM0	REG0LR - TIM0	REG0LR - TIM3		F1h
R240	REG0HR - TIM1	DCPR - TIM0	REG0HR - TIM0	REG0HR - TIM3		F0h

REGISTER DESCRIPTION (Cont'd)

CAPTURE LOAD 0 HIGH REGISTER (REG0HR)

R240 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to capture values from the Up/Down counter or load preset values (MSB).

CAPTURE LOAD 0 LOW REGISTER (REG0LR)

R241 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to capture values from the Up/Down counter or load preset values (LSB).

CAPTURE LOAD 1 HIGH REGISTER (REG1HR)

R242 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to capture values from the Up/Down counter or load preset values (MSB).

CAPTURE LOAD 1 LOW REGISTER (REG1LR)

R243 - Read/Write
Register Page: 10
Reset value: undefined

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to capture values from the Up/Down counter or load preset values (LSB).

COMPARE 0 HIGH REGISTER (CMP0HR)

R244 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to store the MSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 0 LOW REGISTER (CMP0LR)

R245 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to store the LSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 1 HIGH REGISTER (CMP1HR)

R246 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R15	R14	R13	R12	R11	R10	R9	R8

This register is used to store the MSB of the 16-bit value to be compared to the Up/Down counter content.

COMPARE 1 LOW REGISTER (CMP1LR)

R247 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
R7	R6	R5	R4	R3	R2	R1	R0

This register is used to store the LSB of the 16-bit value to be compared to the Up/Down counter content.

REGISTER DESCRIPTION (Cont'd)

TIMER CONTROL REGISTER (TCR)
R248 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
CEN	CCP0	CCMP0	CCL	UDC	UDCS	OF0	CS

Bit 7 = **CEN**: *Counter enable.*
This bit is ANDed with the Global Counter Enable bit (GCEN) in the CICR register (R230). The GCEN bit is set after the Reset cycle.
0: Stop the counter and prescaler
1: Start the counter and prescaler (without reload).

Bit 6 = **CCP0**: *Clear on capture.*
0: No effect
1: Clear the counter and reload the prescaler on a REG0R or REG1R capture event

Bit 5 = **CCMP0**: *Clear on Compare.*
0: No effect
1: Clear the counter and reload the prescaler on a CMP0R compare event

Bit 4 = **CCL**: *Counter clear.*
This bit is reset by hardware after being set by software (this bit always returns "0" when read).
0: No effect
1: Clear the counter without generating an interrupt request

Bit 3 = **UDC**: *Up/Down software selection.*
If the direction of the counter is not fixed by hardware (TxINA and/or TxINB pins, see par. 10.3) it can be controlled by software using the UDC bit.
0: Down counting
1: Up counting

Bit 2 = **UDCS**: *Up/Down count status.*
This bit is read only and indicates the direction of the counter.
0: Down counting
1: Up counting

Bit 1 = **OF0**: *OVF/UNF state.*
This bit is read only.
0: No overflow or underflow occurred
1: Overflow or underflow occurred during a Capture on Register 0

Bit 0 = **CS** *Counter Status.*
This bit is read only and indicates the status of the counter.
0: Counter halted.
1: Counter running.

REGISTER DESCRIPTION (Cont'd)

TIMER MODE REGISTER (TMR)

R249 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
OE1	OE0	BM	RM1	RM0	ECK	REN	C0

Bit 7 = **OE1**: *Output 1 enable.*
0: Disable the Output 1 (TxOUTB pin) and force it high.
1: Enable the Output 1 (TxOUTB pin)
The relevant I/O bit must also be set to Alternate Function

Bit 6 = **OE0**: *Output 0 enable.*
0: Disable the Output 0 (TxOUTA pin) and force it high
1: Enable the Output 0 (TxOUTA pin).
The relevant I/O bit must also be set to Alternate Function

Bit 5 = **BM**: *Bivalue mode*.
This bit works together with the RM1 and RM0 bits to select the timer operating mode (see Table 26 Timer Operating Modes).
0: Disable bivalue mode
1: Enable bivalue mode

Bit 4 = **RM1**: *REG1R mode*.
This bit works together with the BM and RM0 bits to select the timer operating mode. Refer to Table 26 Timer Operating Modes.

Note: This bit has no effect when the Bivalue Mode is enabled (BM=1).

Bit 3 = **RM0**: *REG0R mode*.
This bit works together with the BM and RM1 bits to select the timer operating mode. Refer to Table 26 Timer Operating Modes.

Table 26. Timer Operating Modes

TMR Bits			Timer Operating Modes
BM	RM1	RM0	
1	x	0	Biload mode
1	x	1	Bicapture mode
0	0	0	Load from REG0R and Monitor on REG1R
0	1	0	Load from REG0R and Capture on REG1R
0	0	1	Capture on REG0R and Monitor on REG1R
0	1	1	Capture on REG0R and REG1R

Bit 2 = **ECK** *Timer clock control*.
0: The prescaler clock source is selected depending on the IN0 - IN3 bits in the ICR register
1: Enter Parallel mode (for Timer 1 and Timer 3 only, no effect for Timer 0 and 2). See Section 9.2.2.11.

Bit 1 = **REN**: *Retrigger mode*.
0: Enable retriggerable mode.
1: Disable retriggerable mode

Bit 0 = **CO**: *Continous/One shot mode*.
0: Continuous mode (with autoreload on End of Count condition)
1: One shot mode

REGISTER DESCRIPTION (Cont'd)

EXTERNAL INPUT CONTROL REGISTER (ICR)
R250 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
IN3	IN2	IN1	IN0	A0	A1	B0	B1

Bit 7:4 = **IN[3:0]**: *Input pin function.*
These bits are set and cleared by software.

IN[3:0] bits	TxINA Pin Function	TxINB Input Pin Function
0000	not used	not used
0001	not used	Trigger
0010	Gate	not used
0011	Gate	Trigger
0100	not used	Ext. Clock
0101	Trigger	not used
0110	Gate	Ext. Clock
0111	Trigger	Trigger
1000	Clock Up	Clock Down
1001	Up/Down	Ext. Clock
1010	Trigger Up	Trigger Down
1011	Up/Down	not used
1100	Autodiscr.	Autodiscr.
1101	Trigger	Ext. Clock
1110	Ext. Clock	Trigger
1111	Trigger	Gate

Bit 3:2 = **A[0:1]**: *TxINA Pin event*
These bits are set and cleared by software.

A0	A1	TxINA Pin Event
0	0	No operation
0	1	Falling edge sensitive
1	0	Rising edge sensitive
1	1	Rising and falling edges

Bit 1:0 = **B[0:1]:** *TxINB Pin event*
These bits are set and cleared by software.

B0	B1	TxINB Pin Event
0	0	No operation
0	1	Falling edge sensitive
1	0	Rising edge sensitive
1	1	Rising and falling edges

PRESCALER REGISTER (PRSR)
R251 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
P7	P6	P5	P4	P3	P2	P1	P0

This register holds the preset value for the 8-bit prescaler. The PRSR content may be modified at any time, but it will be loaded into the prescaler at the following prescaler underflow, or as a consequence of a counter reload (either by software or upon external request).

Following a RESET condition, the prescaler is automatically loaded with 00h, so that the prescaler divides by 1 and the maximum counter clock is generated (OSCIN frequency divided by 6 when MODER.5 = DIV2 bit is set).

The binary value programmed in the PRSR register is equal to the divider value minus one. For example, loading PRSR with 24 causes the prescaler to divide by 25.

OUTPUT A CONTROL REGISTER (OACR)
R252 - Read/Write
Register Page: 10
Reset value: 0000 0000

7							0
C0E0	C0E1	C1E0	C1E1	OUE0	OUE1	CEV	0P

Note: Whenever more than one event occurs simultaneously, the action taken will be the result of ANDing the event bits xxE1-xxE0.

Bit 7:6 = **C0E[0:1]**: *COMP0 event bits*.
These bits are set and cleared by software.

C0E0	C0E1	Action on TxOUTA pin on a successful compare of the CMP0R register
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

REGISTER DESCRIPTION (Cont'd)

Bit 5:4 = **C1E[0:1]:** *COMP1 event bits.*
These bits are set and cleared by software.

C1E0	C1E1	Action on TxOUTA pin on a successful compare of the CMP1R register
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Bit 3:2 = **OUE[0:1]**: *OVF/UNF event bits.*
These bits are set and cleared by software.

OUE0	OUE1	Action on TxOUTA pin on an Overflow or Underflow on the U/D counter
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Note: Whenever more than one event occurs simultaneously, the action taken will be the result of ANDing the event xxE1-xxE0 bits.

Bit 1 = **CEV**: *On-Chip event on CMP0R*
This bit is set and cleared by software.
0: No action
1: A successful compare on CMP0R activates the on-chip event signal (a single pulse is generated)

Bit 0 = **OP**: *TxOUTA preset value*
This bit is set and cleared by software and by hardware. The value of this bit is the preset value of the TxOUTA pin. Reading this bit returns the current state of the TxOUTA pin (useful when it is selected in toggle mode).

OUTPUT B CONTROL REGISTER (OBCR)
R253 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
C0E0	C0E1	C1E0	C1E1	OUE0	OUE1	OEV	0P

Note: Whenever more than one event occurs simultaneously, the action taken will be the result of ANDing the event bits xxE1-xxE0.

Bit 7:6 = **C0E[0:1]**: *COMP0 event bits.*
These bits are set and cleared by software.

C0E0	C0E1	Action on TxOUTB pin on a successful compare of the CMP0R register
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Bit 5:4 = **C1E[0:1]:** *COMP1 event bits.*
These bits are set and cleared by software.

C1E0	C1E1	Action on TxOUTB pin on a successful compare of the CMP1R register
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Bit 3:2 = **OUE[0:1]**: *OVF/UNF event bits.*
These bits are set and cleared by software.

OUE0	OUE1	Action on TxOUTB pin on an Overflow or Underflow on the U/D counter
0	0	Set
0	1	Toggle
1	0	Reset
1	1	NOP

Bit 1 = **OEV**: *On-Chip event on OVF/UNF.*
This bit is set and cleared by software.
0: No action
1: An underflow/overflow activates the on-chip event signal (a single pulse is generated)

REGISTER DESCRIPTION (Cont'd)

Bit 0 = **OP**: *TxOUTB preset value*
This bit is set and cleared by software and by hardware. The value of this bit is the preset value of the TxOUTB pin. Reading this bit returns the current state of the TxOUTB pin (useful when it is selected in toggle mode).

FLAG REGISTER (FLAGR)
R254 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
CP0	CP1	CM0	CM1	OUF	OCP0	OCM0	A0

Bit 7 = **CP0**: *Capture 0 flag.*
This bit is set by hardware after a capture on REG0R register. An interrupt is generated depending on the value of the GTIEN, CP0I bits in the IDMR register and the A0 bit in the FLAGR register. The CP0 bit must be cleared by software. Setting by software acts as a software load/capture to/from the REG0R register.
0: No Capture 0 event
1: Capture 0 event occurred

Bit 6 = **CP1**: *Capture 1 flag.*
This bit is set by hardware after a capture on REG1R register. An interrupt is generated depending on the value of the GTIEN, CP0I bits in the IDMR register and the A0 bit in the FLAGR register. The CP1 bit must be cleared by software. Setting by software acts as a software capture on the REG1R register, except when in Bicapture mode.
0: No Capture 1 event
1: Capture 1 event occurred

Bit 5 = **CM0**: *Compare 0 flag.*
This bit is set by hardware after a successful compare on the CMP0R register. An interrupt is generated if the GTIEN and CM0I bits in the IDMR register are set. The CM0 bit is cleared by software.
0: No Compare 0 event
1: Compare 0 event occurred

Bit 4 = **CM1**: *Compare 1 flag.*
This bit is set after a successful compare on CMP1R register. An interrupt is generated if the GTIEN and CM1I bits in the IDMR register are set. The CM1 bit is cleared by software.
0: No Compare 1 event
1: Compare 1 event occurred

Bit 3 = **OUF**: *Overflow/Underflow.*
This bit is set by hardware after a counter Over/Underflow condition. An interrupt is generated if GTIEN and OUI=1 in the IDMR register. The OUF bit is cleared by software.
0: No counter overflow/underflow
1: Counter overflow/underflow

Bit 2 = **OCP0**: *Overrun on Capture 0.*
This bit is set by hardware when more than one INT/DMA requests occur before the CP0 flag is cleared by software or whenever a capture is simulated by setting the CP0 flag by software. The OCP0 flag is cleared by software.
0: No capture 0 overrun
1: Capture 0 overrun

Bit 1 = **OCM0**: *Overrun on compare 0.*
This bit is set by hardware when more than one INT/DMA requests occur before the CM0 flag is cleared by software.The OCM0 flag is cleared by software.
0: No compare 0 overrun
1: Compare 0 overrun

Bit 0 = **A0**: *Capture interrupt function*
This bit is set and cleared by software.
0: Configure the capture interrupt as an OR function of REG0R/REG1R captures
1: Configure the capture interrupt as an AND function of REG0R/REG1R captures

REGISTER DESCRIPTION (Cont'd)

INTERRUPT/DMA MASK REGISTER (IDMR)
R254 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
GTIEN	CP0D	CP0I	CP1I	CM0D	CM0I	CM1I	OUI

Bit 7 = **GTIEN**: *Global timer interrupt enable.*
This bit is set and cleared by software.
0: Disable all Timer interrupts
1: Enable all timer Timer Interrupts from enabled sources

Bit 6 = **CP0D**: *Capture 0 DMA mask.*
0: Disable capture on REG0R DMA
1: Enable capture on REG0R DMA

Bit 5 = **CP0I**: *Capture 0 interrupt mask*
0: Disable capture on REG0R interrupt
1: Enable capture on REG0R interrupt

Bit 4 = **CP1I**: *Capture 1 interrupt mask*
This bit is set and cleared by software.
0: Disable capture on REG1R interrupt
1: Enable capture on REG1R interrupt

Bit 3 = **CM0D**: *Compare 0 DMA mask.*
This bit is set and cleared by software.
0: Disable compare on CMP0R DMA
1: Enable compare on CMP0R DMA

Bit 2 = **CM0I**: *Compare 0 Interrupt mask*
This bit is set and cleared by software.
0: Disable compare on CMP0R interrupt
1: Enable compare on CMP0R interrupt

Bit 1 = **CM1I**: *Compare 1 Interrupt mask*
This bit is set and cleared by software.
0: Disable compare on CMP1R interrupt
1: Enable compare on CMP1R interrupt

Bit 0 = **OUI**:
Overflow/Underflow interrupt mask
This bit is set and cleared by software.
0: Disable Overflow/Underflow interrupt
1: Enable Overflow/Underflow interrupt

DMA COUNTER POINTER REGISTER (DCPR)
R240 - Read/Write
Register Page: 9
Reset value: undefined

7							0
DCP7	DCP6	DCP5	DCP4	DCP3	DCP2	DMA SRCE	REG/ MEM

Bit 7:2 = **DCP[7:2]**: *MSBs of DMA counter register address.*
These are the most significant bits of the DMA counter register address programmable by software. The DCP2 bit may also be toggled by hardware if the Timer DMA section for the Compare 0 channel is configured in Swap mode.

Bit 1 = **DMA-SRCE**: *DMA source selection.*
This bit is set and cleared by hardware.
0: DMA source is a Capture on REG0R register
1: DMA destination is a Compare on CMP0R register

Bit 0 = **REG/MEM**: *DMA area selection.*
This bit is set and cleared by software. It selects the source and destination of the DMA area
0: DMA from/to memory
1: DMA from/to Register File

DMA ADDRESS POINTER REGISTER (DAPR)
R241 - Read/Write
Register Page: 9
Reset value: undefined

7							0
DAP7	DAP6	DAP5	DAP4	DAP3	DAP2	DMA SRCE	PRG /DAT

Bit 7:2 = **DAP[7:2]**: *MSB of DMA address register location.*
These are the most significant bits of the DMA address register location programmable by software. The DAP2 bit may also be toggled by hardware if the Timer DMA section for the Compare 0 channel is configured in Swap mode.

Note: During a DMA transfer with the Register File, the DAPR is not used; however, in Swap mode, DAPR(2) is used to point to the correct table.

REGISTER DESCRIPTION (Cont'd)

Bit 1 = **DMA-SRCE**: *DMA source selection.*
This bit is fixed by hardware.
0: DMA source is a Capture on REG0R register
1: DMA destination is a Compare on the CMP0R register

Bit 0 = **PRG/DAT:** *DMA memory selection.*
Bit 0 = **DP**: *Memory Segment Selector.*
This bit is set and cleared by software. It is only meaningful if DAPR.REG/MEM=0.
0: The ISR register is used to extend the address of data transferred by DMA (see MMU chapter).
1: The DMASR register is used to extend the address of data transferred by DMA (see MMU chapter).

REG/MEM	PRG/DAT	DMA Source/Destination
0	0	ISR register used to address memory
0	1	DMASR register used to address memory
1	0	Register file
1	1	Register file

INTERRUPT VECTOR REGISTER (IVR)

R242 - Read/Write
Register Page: 9
Reset value: **xxxx xxx0**

7							0
V4	V3	V2	V1	V0	W1	W0	0

This register is used as a vector, pointing to the 16-bit interrupt vectors in memory which contain the starting addresses of the three interrupt subroutines managed by each timer.

Only one Interrupt Vector Register is available for each timer, and it is able to manage three interrupt groups, because the 3 least significant bits are fixed by hardware depending on the group which generated the interrupt request.

In order to determine which request generated the interrupt within a group, the FLAGR register can be used to check the relevant interrupt source.

Bit 7:3 = **V[4:0]:** *MSB of the vector address.*
These bits are user programmable and contain the five most significant bits of the Timer interrupt vector addresses in memory. In any case, an 8-bit address can be used to indicate the Timer interrupt vector locations, because they are within the first 256 memory locations (see Interrupt and DMA chapters).

Bit 2:1 = **W[1:0]:** *Vector address bits.*
These bits are equivalent to bit 1 and bit 2 of the Timer interrupt vector addresses in memory. They are fixed by hardware, depending on the group of sources which generated the interrupt request as follows:.

W1	W0	Interrupt Source
0	0	Overflow/Underflow even interrupt
0	1	Not available
1	0	Capture event interrupt
1	1	Compare event interrupt

Bit 0 = This bit is forced by hardware to 0.

INTERRUPT/DMA CONTROL REGISTER (IDCR)

R243 - Read/Write
Register Page: 9
Reset value: 1100 0111 (C7h)

7							0
CPE	CME	DCTS	DCTD	SWEN	PL2	PL1	PL0

Bit 7 = **CPE**: *Capture 0 EOB.*
This bit is set by hardware when the End Of Block condition is reached during a Capture 0 DMA operation with the Swap mode enabled. When Swap mode is disabled (SWEN bit = "0"), the CPE bit is forced to 1 by hardware.
0: No end of block condition
1: Capture 0 End of block

Bit 6 = **CME**: *Compare 0 EOB.*
This bit is set by hardware when the End Of Block condition is reached during a Compare 0 DMA operation with the Swap mode enabled. When the Swap mode is disabled (SWEN bit = "0"), the CME bit is forced to 1 by hardware.
0: No end of block condition
1: Compare 0 End of block

REGISTER DESCRIPTION (Cont'd)

Bit 5 = **DCTS**: *DMA capture transfer source.*
This bit is set and cleared by software. It selects the source of the DMA operation related to the channel associated with the Capture 0.
Note: The I/O port source is available only on specific devices.
0: REG0R register
1: I/O port.

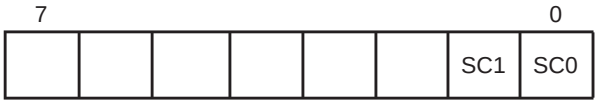
Bit 4 = **DCTD**: *DMA compare transfer destination*
This bit is set and cleared by software. It selects the destination of the DMA operation related to the channel associated with Compare 0.
Note: The I/O port destination is available only on specific devices.
0: CMP0R register
1: I/O port

Bit 3 = **SWEN**: *Swap function enable.*
This bit is set and cleared by software.
0: Disable Swap mode
1: Enable Swap mode for both DMA channels.

Bit 2:0 = **PL[2:0]:** *Interrupt/DMA priority level.*
With these three bits it is possible to select the Interrupt and DMA priority level of each timer, as one of eight levels (see Interrupt/DMA chapter).

I/O CONNECTION REGISTER (IOCR)

R248 - Read/Write
Register Page: 9
Reset value: 1111 1100 (FCh)



Bit 7:2 = not used.

Bit 1 = **SC1**: *Select connection odd.*
This bit is set and cleared by software. It selects if the TxOUTA and TxINA pins for Timer 1 and Timer 3 are connected on-chip or not.
0: T1OUTA / T1INA and T3OUTA/ T3INA unconnected
1: T1OUTA connected internally to T1INA and T3OUTA connected internally to T3INA

Bit 0 = **SC0**: *Select connection even.*
This bit is set and cleared by software. It selects if the TxOUTA and TxINA pins for Timer 0 and Timer 2 are connected on-chip or not.
0: T0OUTA / T0INA and T2OUTA/ T2INA unconnected
1: T0OUTA connected internally to T0INA and T2OUTA connected internally to T2INA

9.3 STANDARD TIMER (STIM)

9.3.1 Introduction

The Standard Timer includes a programmable 16-bit down counter and an associated 8-bit prescaler with Single and Continuous counting modes capability. The Standard Timer uses an input pin (STIN) and an output (STOUT) pin. These pins, when available, may be independent pins or connected as Alternate Functions of an I/O port bit.

STIN can be used in one of four programmable input modes:

– event counter,

– gated external input mode,

– triggerable input mode,

– retriggerable input mode.

STOUT can be used to generate a Square Wave or Pulse Width Modulated signal.

The Standard Timer is composed of a 16-bit down counter with an 8-bit prescaler. The input clock to the prescaler can be driven either by an internal clock equal to INTCLK divided by 4, or by CLOCK2 derived directly from the external oscillator, divided by device dependent prescaler value, thus providing a stable time reference independent from the PLL programming or by an external clock connected to the STIN pin.

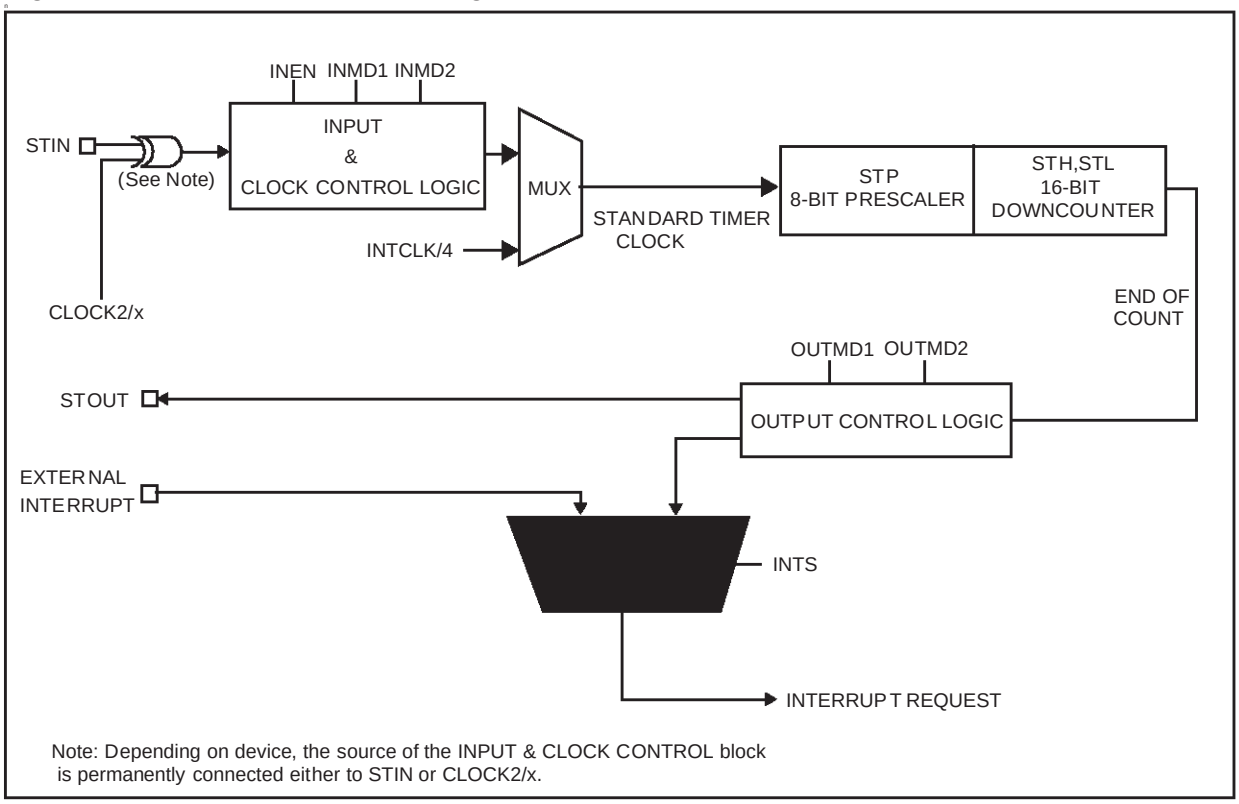
The Standard Timer End Of Count condition is able to generate an interrupt which is connected to one of the external interrupt channels.

The End of Count condition is defined as the Counter Underflow, whenever 00h is reached.

9.3.1.1 Device-Specific Options

Depending on the ST9 variant and package type, some STIM interface signals described in this chapter may not be connected to an external pin. Refer to the device pinout description.

Figure 61. Standard Timer Block Diagram



Note: Depending on device, the source of the INPUT & CLOCK CONTROL block is permanently connected either to STIN or CLOCK2/x.

STANDARD TIMER (Cont'd)

9.3.2 Functional Description

9.3.2.1 Timer/Counter control

Start-stop Count. The ST-SP bit (STC.7) is used in order to start and stop counting. An instruction which sets this bit will cause the Standard Timer to start counting at the beginning of the next instruction. Resetting this bit will stop the counter.

If the counter is stopped and restarted, counting will resume from the value held at the stop condition, unless a new constant has been entered in the Standard Timer registers during the stop period. In this case, the new constant will be loaded as soon as counting is restarted.

A new constant can be written in STH, STL, STP registers while the counter is running. The new value of the STH and STL registers will be loaded at the next End of Count condition, while the new value of the STP register will be loaded immediately.

WARNING: In order to prevent incorrect counting of the Standard Timer, the prescaler (STP) and counter (STL, STH) registers must be initialised before the starting of the timer. If this is not done, counting will start with the reset values (STH=FFh, STL=FFh, STP=FFh).

Single/Continuous Mode.
The S-C bit (STC.6) selects between the Single or Continuous mode.

SINGLE MODE: at the End of Count, the Standard Timer stops, reloads the constant and resets the Start/Stop bit (the user programmer can inspect the timer current status by reading this bit). Setting the Start/Stop bit will restart the counter.

CONTINUOUS MODE: At the End of the Count, the counter automatically reloads the constant and restarts. It is only stopped by resetting the Start/Stop bit.

The S-C bit can be written either with the timer stopped or running. It is possible to toggle the S-C bit and start the Standard Timer with the same instruction.

9.3.2.2 Standard Timer Input Modes (ST9 devices with Standard Timer Input STIN)

Bits INMD2, INMD1 and INEN are used to select the input modes. The Input Enable (INEN) bit enables the input mode selected by the INMD2 and INMD1 bits. If the input is disabled (INEN="0"), the values of INMD2 and INMD1 are not taken into account. In this case, this unit acts as a 16-bit timer (plus prescaler) directly driven by INTCLK/4 and transitions on the input pin have no effect.

Event Counter Mode (INMD1 = "0", INMD2 = "0")

The Standard Timer is driven by the signal applied to the input pin (STIN) which acts as an external clock. The unit works therefore as an event counter. The event is a high to low transition on STIN. Spacing between trailing edges should be at least the period of INTCLK multiplied by 8 (i.e. the maximum Standard Timer input frequency is 2.5 MHz with INTCLK = 20MHz).

Gated Input Mode (INMD1 = "0", INMD2 = "1")

The Timer uses the internal clock (INTCLK divided by 4) and starts and stops the Timer according to the state of STIN pin. When the status of the STIN is High the Standard Timer count operation proceeds, and when Low, counting is stopped.

Triggerable Input Mode (INMD1 = "1", INMD2 = "0")

The Standard Timer is started by:

a) setting the Start-Stop bit, AND

b) a High to Low (low trigger) transition on STIN.

In order to stop the Standard Timer in this mode, it is only necessary to reset the Start-Stop bit.

Retriggerable Input Mode (INMD1 = "1", INMD2 = "1")

In this mode, when the Standard Timer is running (with internal clock), a High to Low transition on STIN causes the counting to start from the last constant loaded into the STL/STH and STP registers. When the Standard Timer is stopped (ST-SP bit equal to zero), a High to Low transition on STIN has no effect.

9.3.2.3 Time Base Generator (ST9 devices without Standard Timer Input STIN)

For devices where STIN is replaced by a connec-

STANDARD TIMER (Cont'd)

9.3.2.4 Standard Timer Output Modes

OUTPUT modes are selected using 2 bits of the STC register: OUTMD1 and OUTMD2.

No Output Mode (OUTMD1 = "0", OUTMD2 = "0")

With this setting, the Standard Timer output is disabled and the output pin is held at a "1" level to allow several alternate functions on the same pin.

Square Wave Output Mode (OUTMD1 = "0", OUTMD2 = "1")

The Standard Timer toggles the state of the STOUT pin on every End Of Count condition. With INTCLK = 12MHz, this allows generation of a square wave with a period ranging from 666ns to 11.18 seconds.

PWM Output Mode (OUTMD1 = "1")

The value of the OUTMD2 bit is transferred to the STOUT output pin at the End Of Count. This allows the user to generate PWM signals, by modifying the status of OUTMD2 between End of Count events, based on software counters decremented on the Standard Timer interrupt.

9.3.3 Interrupt Selection

The Standard Timer may generate an interrupt request at every End of Count.

Bit 2 of the STC register (INTS) selects the interrupt source between the Standard Timer interrupt and the external interrupt pin. Thus the Standard Timer Interrupt uses the interrupt channel and takes the priority and vector of the external interrupt channel.

If INTS is set to "1", the Standard Timer interrupt is disabled; otherwise, an interrupt request is generated at every End of Count.

Note: When enabling or disabling the Standard Timer Interrupt (writing INTS in the STC register) an edge may be generated on the interrupt channel, causing an unwanted interrupt.

To avoid this spurious interrupt request, the INTS bit should be accessed only when the interrupt logic is disabled (i.e. after the DI instruction). It is also necessary to clear any possible interrupt pending requests on the corresponding external interrupt channel before enabling it. A delay instruction (i.e. a NOP instruction) must be inserted between the reset of the interrupt pending bit and the INTS write instruction.

9.3.4 Register Mapping

The ST9 can have up to 4 Standard Timers.

Each Standard Timer has 4 registers mapped into Page 11 in Group F of the Register File

In the register description on the following page, register addresses refer to STIM0 only.

STD Timer	Register	Register Address
STIM0	STH0	R240 (F0h)
	STL0	R241 (F1h)
	STP0	R242 (F2h)
	STC0	R243 (F3h)
STIM1	STH1	R244 (F4h)
	STL1	R245 (F5h)
	STP1	R246 (F6h)
	STC1	R247 (F7h)
STIM2	STH2	R248 (F8h)
	STL2	R249 (F9h)
	STP2	R250 (FAh)
	STC2	R251 (FBh)
STIM3	STH3	R252 (FCh)
	STL3	R253 (FDh)
	STP3	R254 (FEh)
	STC3	R255 (FFh)

9.3.5 Register Description

COUNTER HIGH BYTE REGISTER(STH)
R240 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
ST.15	ST.14	ST.13	ST.12	ST.11	ST.10	ST.9	ST.8

Bit 7:0 = **ST.[15:8]**: *Counter High-Byte.*

COUNTER LOW BYTE REGISTER(STL)
R241 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
ST.7	ST.6	ST.5	ST.4	ST.3	ST.2	ST.1	ST.0

Bit 7:0 = **ST.[7:0]**: *Counter Low Byte.*

Writing to the STH and STL registers allows the user to enter the Standard Timer constant, while reading it provides the counter's current value. Thus it is possible to read the counter on-the-fly.

STANDARD TIMER PRESCALER REGISTER (STP)
R242 - Read/Write
Register Page: 11
Reset value: 1111 1111 (FFh)

7							0
STP.7	STP.6	STP.5	STP.4	STP.3	STP.2	STP.1	STP.0

Bit 7:0 = **STP.[7:0]**: *Prescaler.*
The Prescaler value for the Standard Timer is programmed into this register. When reading the STP register, the returned value corresponds to the programmed data instead of the current data.
00h: No prescaler
01h: Divide by 2
FFh: Divide by 256

STANDARD TIMER CONTROL REGISTER (STC)
R243 - Read/Write
Register Page: 11
Reset value: 0001 0100 (14h)

7							0
ST-SP	S-C	INMD1	INMD2	INEN	INTS	OUTMD1	OUTMD2

Bit 7 = **ST-SP**: *Start-Stop Bit.*
This bit is set and cleared by software.
0: Stop counting
1: Start counting

Bit 6 = **S-C**: *Single-Continuous Mode Select.*
This bit is set and cleared by software.
0: Continuous Mode
1: Single Mode

Bit 5:4 = **INMD[1:2]**: *Input Mode Selection.*
These bits select the Input functions as shown in Section 9.3.2.2, when enabled by INEN.

INMD1	INMD2	Mode
0	**0**	Event Counter mode
0	**1**	Gated input mode
1	**0**	Triggerable mode
1	**1**	Retriggerable mode

Bit 3 = **INEN**: *Input Enable.*
This bit is set and cleared by software. If neither the STIN pin nor the CLOCK2 line are present, INEN must be 0.
0: Input section disabled
1: Input section enabled

Bit 2 = **INTS**: *Interrupt Selection.*
0: Standard Timer interrupt enabled
1: Standard Timer interrupt is disabled and the external interrupt pin is enabled.

Bit 1:0 = **OUTMD[1:2]**: Output Mode Selection.
These bits select the output functions as described in Section 9.3.2.4. .

OUTMD1	OUTMD2	Mode
0	**0**	No output mode
0	**1**	Square wave output mode
1	**x**	PWM output mode

9.4 SERIAL PERIPHERAL INTERFACE (SPI)

9.4.1 Introduction

The Serial Peripheral Interface (SPI) is a general purpose on-chip shift register peripheral. It allows communication with external peripherals via an SPI protocol bus.

In addition, special operating modes allow reduced software overhead when implementing I^2C-bus and IM-bus communication standards.

The SPI uses up to 3 pins: Serial Data In (SDI), Serial Data Out (SDO) and Synchronous Serial Clock (SCK). Additional I/O pins may act as device selects or IM-bus address identifier signals.

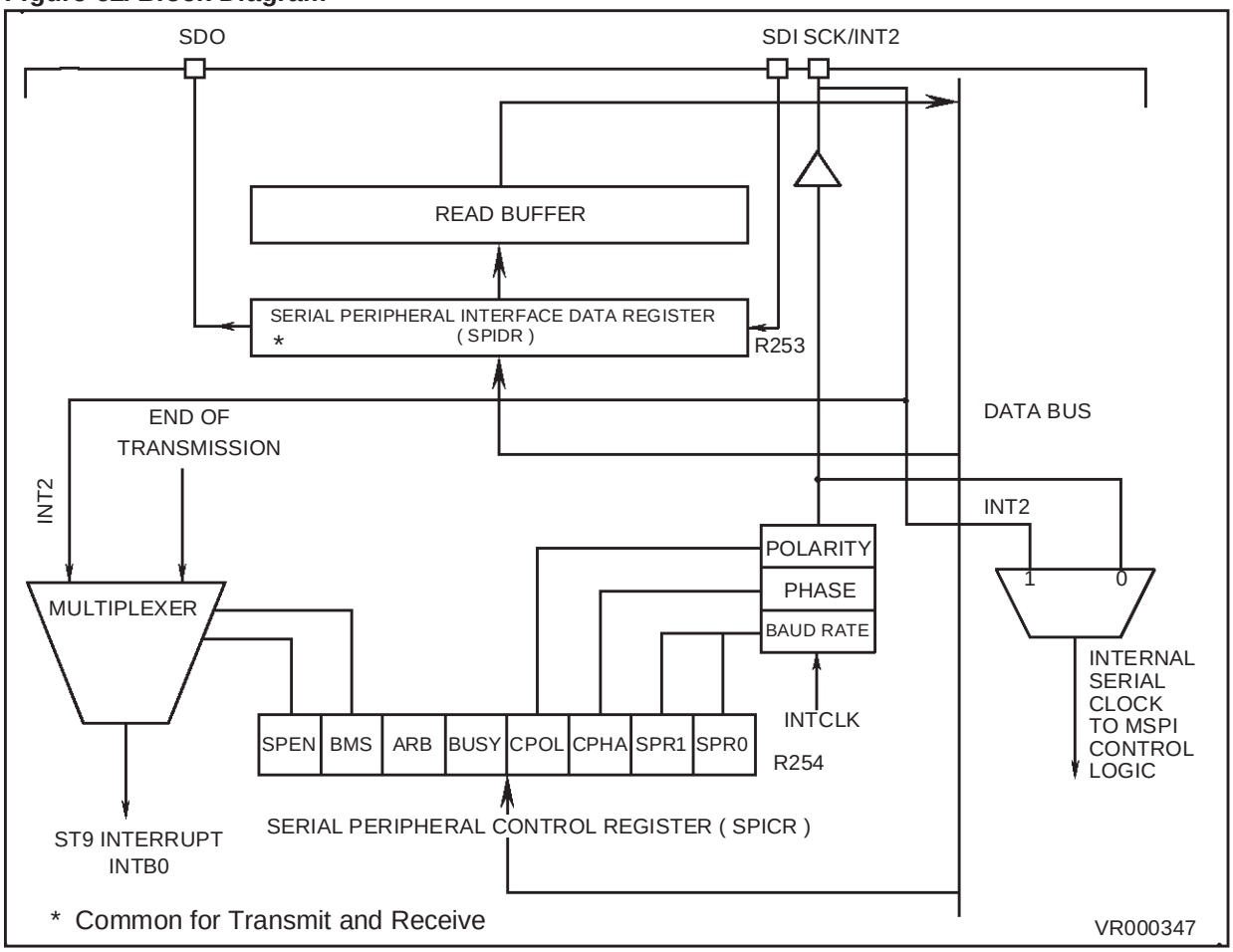
The main features are:

- Full duplex synchronous transfer if 3 I/O pins are used
- Master operation only
- 4 Programmable bit rates
- Programmable clock polarity and phase
- Busy Flag
- End of transmission interrupt
- Additional hardware to facilitate more complex protocols

9.4.2 Device-Specific Options

Depending on the ST9 variant and package type, the SPI interface signals may not be connected to separate external pins. Refer to the Peripheral Configuration Chapter for the device pin-out.

Figure 62. Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

9.4.3 Functional Description

The SPI, when enabled, receives input data from the internal data bus to the SPI Data Register (SPIDR). A Serial Clock (SCK) is generated by controlling through software two bits in the SPI Control Register (SPICR). The data is parallel loaded into the 8 bit shift register during a write cycle. This is shifted out serially via the SDO pin, MSB first, to the slave device, which responds by sending its data to the master device via the SDI pin. This implies full duplex transmission if 3 I/O pins are used with both the data-out and data-in synchronized with the same clock signal, SCK. Thus the transmitted byte is replaced by the received byte, eliminating the need for separate "Tx empty" and "Rx full" status bits.

When the shift register is loaded, data is parallel transferred to the read buffer and becomes available to the CPU during a subsequent read cycle.

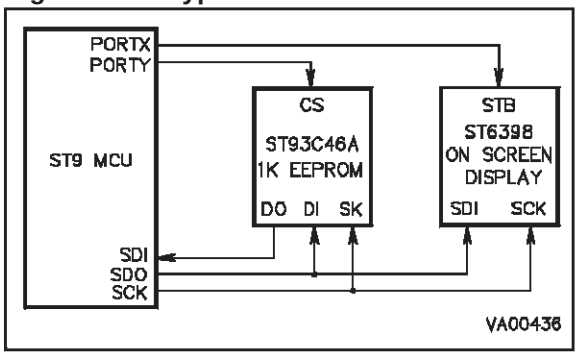
The SPI requires three I/O port pins:

SCK	Serial Clock signal
SDO	Serial Data Out
SDI	Serial Data In

An additional I/O port output bit may be used as a slave chip select signal. Data and Clock pins I C Bus protocol are open-drain to allow arbitration and multiplexing.

Figure 63 below shows a typical SPI network.

Figure 63. A Typical SPI Network



9.4.3.1 Input Signal Description

Serial Data In (SDI)

Data is transferred serially from a slave to a master on this line, most significant bit first. In an S-BUS/I^2C-bus configuration, the SDI line senses the value forced on the data line (by SDO or by another peripheral connected to the S-bus/I^2C-bus).

9.4.3.2 Output Signal Description

Serial Data Out (SDO)

The SDO pin is configured as an output for the master device. This is obtained by programming the corresponding I/O pin as an output alternate function. Data is transferred serially from a master to a slave on SDO, most significant bit first. The master device always allows data to be applied on the SDO line one half cycle before the clock edge, in order to latch the data for the slave device. The SDO pin is forced to high impedance when the SPI is disabled.

During an S-Bus or I^2C-Bus protocol, when arbitration is lost, SDO is set to one (thus not driving the line, as SDO is configured as an open drain).

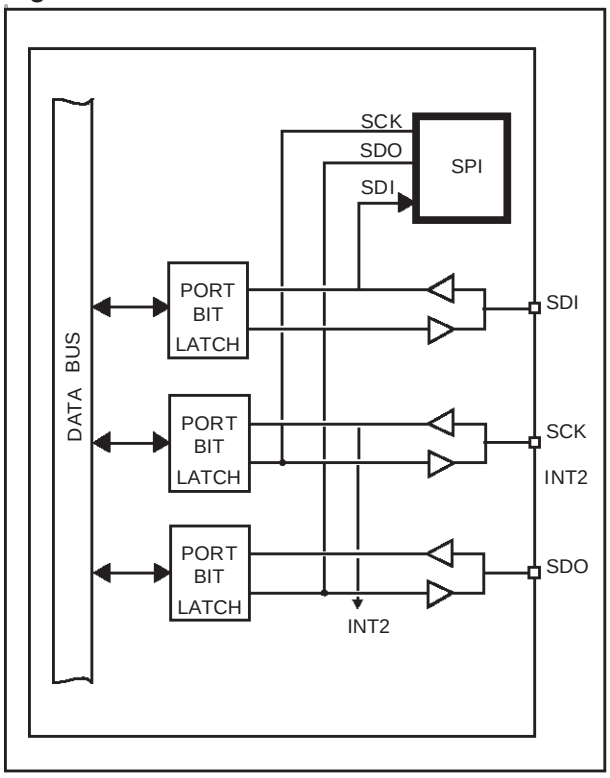
Master Serial Clock (SCK)

The master device uses SCK to latch the incoming data on the SDI line. This pin is forced to a high impedance state when SPI is disabled (SPEN, SPICR.7 = "0"), in order to avoid clock contention from different masters in a multi-master system. The master device generates the SCK clock from INTCLK. The SCK clock is used to synchronize data transfer, both in to and out of the device, through its SDI and SDO pins. The SCK clock type, and its relationship with data is controlled by the CPOL (Clock Polarity) and CPHA (Clock Phase) bits in the Serial Peripheral Control Register (SPICR). This input is provided with a digital filter which eliminates spikes lasting less than one INTCLK period.

Two bits, SPR1 and SPR0, in the Serial Peripheral Control Register (SPICR), select the clock rate. Four frequencies can be selected, two in the high frequency range (mostly used with the SPI protocol) and two in the medium frequency range (mostly used with more complex protocols).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 64. SPI I/O Pins



9.4.4 Interrupt Structure

The SPI peripheral is associated with external interrupt channel B0 (pin INT2). Multiplexing between the external pin and the SPI internal source is controlled by the SPEN and BMS bits, as shown in Table 27 Interrupt Configuration.

The two possible SPI interrupt sources are:

– End of transmission (after each byte).

– S-bus/I^2C-bus start or stop condition.

Care should be taken when toggling the SPEN and/or BMS bits from the "0,0" condition. Before changing the interrupt source from the external pin to the internal function, the B0 interrupt channel should be masked. EIMR.2 (External Interrupt Mask Register, bit 2, IMBO) and EIPR.2 (External Interrupt Pending Register bit 2, IMP0) should be "0" before changing the source. This sequence of events is to avoid the generating and reading of spurious interrupts.

A delay instruction lasting at least 4 clock cycles (e.g. 2 NOPs) should be inserted between the SPEN toggle instruction and the Interrupt Pending bit reset instruction.

The INT2 input Function is always mapped together with the SCK input Function, to allow Start/Stop bit detection when using S-bus/I^2C-bus protocols.

A start condition occurs when SDI goes from "1" to "0" and SCK is "1". The Stop condition occurs when SDI goes from "0" to "1" and SCK is "1". For both Stop and Start conditions, SPEN = "0" and BMS = "1".

Table 27. Interrupt Configuration

SPEN	BMS	Interrupt Source
0	0	External channel INT2
0	1	S-bus/I^2C bus start or stop condition
1	X	End of a byte transmission

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.4.5 Working With Other Protocols

The SPI peripheral offers the following facilities for operation with S-bus/I^2C-bus and IM-bus protocols:

- Interrupt request on start/stop detection
- Hardware clock synchronisation
- Arbitration lost flag with an automatic set of data line

Note that the I/O bit associated with the SPI should be returned to a defined state as a normal I/O pin before changing the SPI protocol.

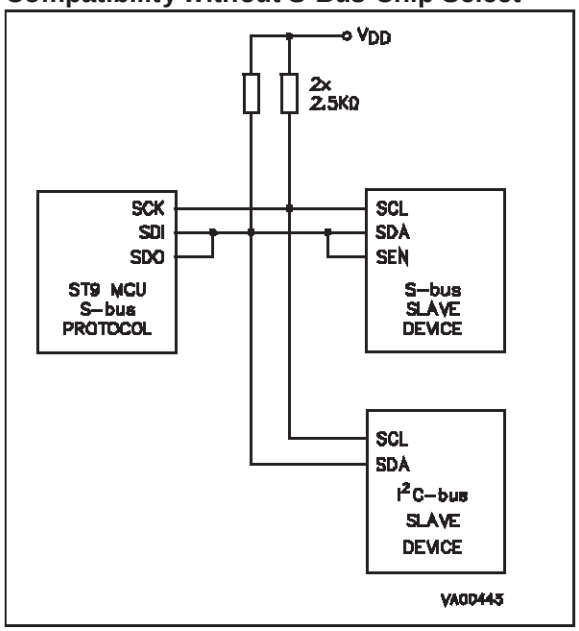
The following paragraphs provide information on how to manage these protocols.

9.4.6 I^2C-bus Interface

The I^2C-bus is a two-wire bidirectional data-bus, the two lines being SDA (Serial DAta) and SCL (Serial CLock). Both are open drain lines, to allow arbitration. As shown in Figure 66, data is toggled with clock low. An I C bus start condition is the transition on SDI from 1 to 0 with the SCK held high. In a stop condition, the SCK is also high and the transition on SDI is from 0 to 1. During both of these conditions, if SPEN = 0 and BMS = 1 then an interrupt request is performed.

Each transmission consists of nine clock pulses (SCL line). The first 8 pulses transmit the byte (MSB first), the ninth pulse is used by the receiver to acknowledge.

Figure 65. S-Bus / I^2C-bus Peripheral Compatibility without S-Bus Chip Select

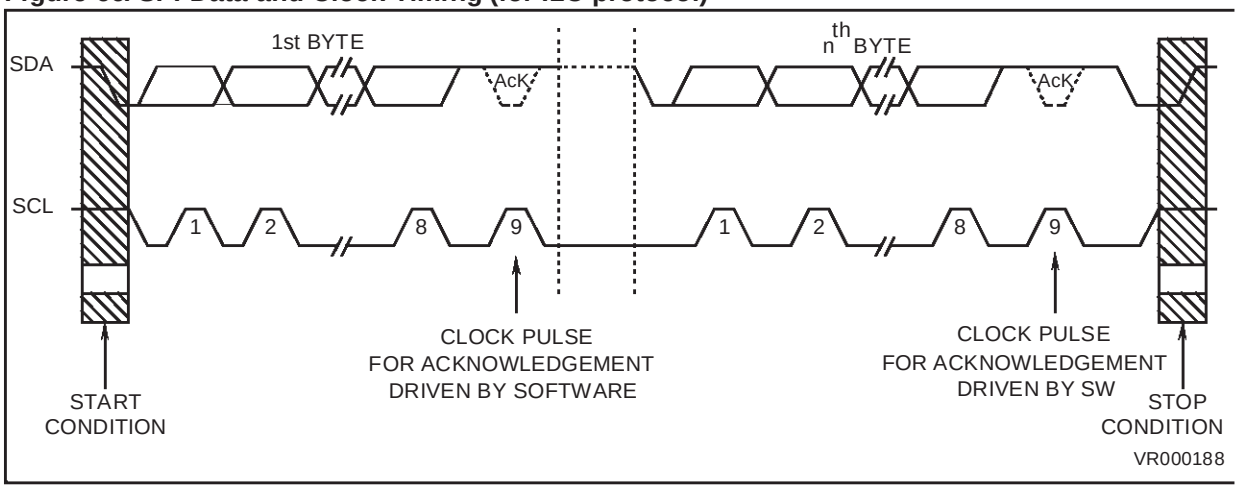


SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 28. Typical I^2C-bus Sequences

Phase	Software	Hardware	Notes
INITIALIZE	SPICR.CPOL, CPHA = 0, 0 SPICR.SPEN = 0 SPICR.BMS = 1 SCK pin set as AF output SDI pin set as input Set SDO port bit to 1	SCK, SDO in HI-Z SCL, SDA = 1, 1	Set polarity and phase SPI disable START/STOP interrupt Enable
START	SDO pin set as output Open Drain Set SDO port bit to 0	SDA = 0, SCL = 1 interrupt request	START condition receiver START detection
TRANSMISSION	SPICR.SPEN = 1 SDO pin as Alternate Function output load data into SPIDR	SCL = 0 Start transmission Interrupt request at end of byte transmission	Managed by interrupt routine load FFh when receiving end of transmission detection
ACKNOWLEDGE	SPICR.SPEN = 0 Poll SDA line Set SDA line SPICR.SPEN = 1	SCK, SDO in HI-Z SCL, SDA = 1 SCL = 0	SPI disable only if transmitting only if receiving only if transmitting
STOP	SDO pin set as output Open Drain SPICR.SPEN = 0 Set SDO port bit to 1	SDA = 1 interrupt request	STOP condition

Figure 66. SPI Data and Clock Timing (for I2C protocol)



SERIAL PERIPHERAL INTERFACE (Cont'd)

The data on the SDA line is sampled on the low to high transition of the SCL line.

SPI working with an I2C-bus

To use the SPI with the I²C-bus protocol, the SCK line is used as SCL; the SDI and SDO lines, externally wire-ORed, are used as SDA. All output pins must be configured as open drain (seeFigure 65).

Figure 28 illustrates the typical I²C-bus sequence, comprising 5 phases: Initialization, Start, Transmission, Acknowledge and Stop. It should be noted that only the first 8 bits are handled by the SPI peripheral; the ACKNOWLEDGE bit must be managed by software, by polling or forcing the SCL and SDO lines via the corresponding I/O port bits.

During the transmission phase, the following I²C-bus features are also supported by hardware.

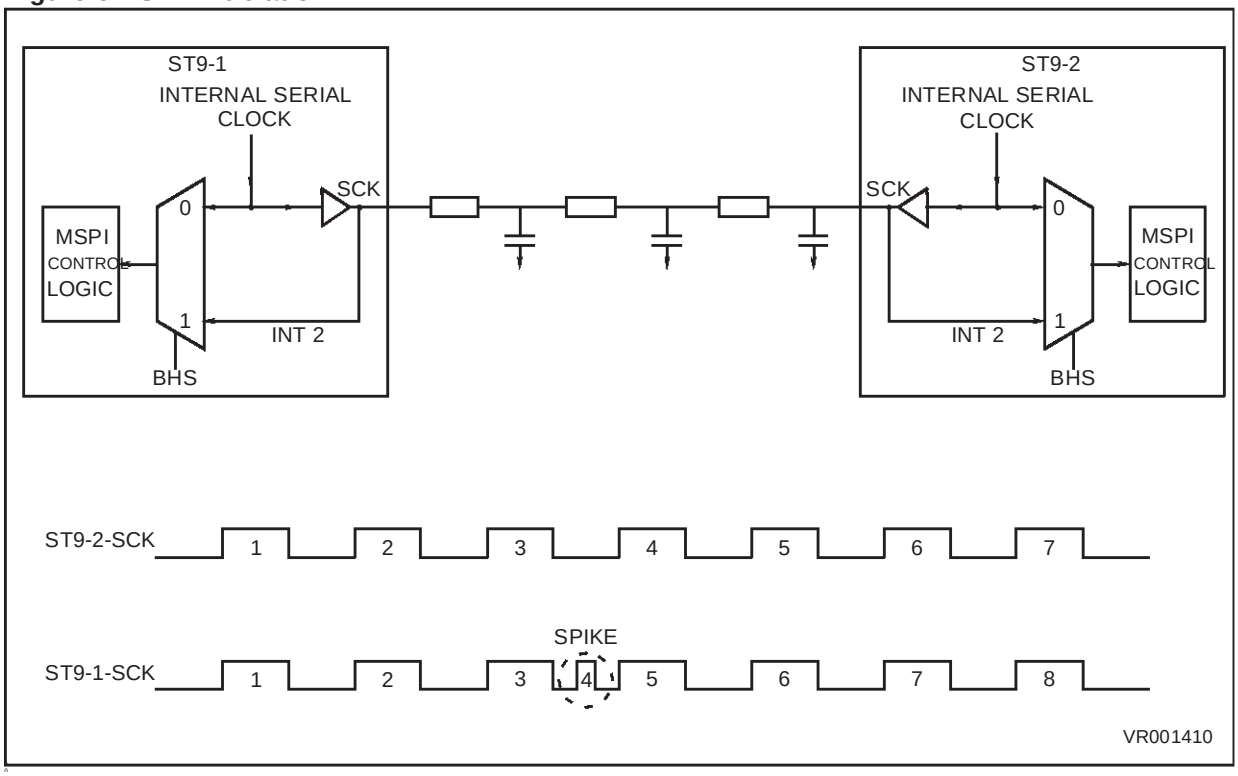
Clock Synchronization

In a multimaster I²C-bus system, when several masters generate their own clock, synchronization is required. The first master which releases the SCL line stops internal counting, restarting only when the SCL line goes high (released by all the other masters). In this manner, devices using different clock sources and different frequencies can be interfaced.

Arbitration Lost

When several masters are sending data on the SDA line, the following takes place: if the transmitter sends a "1" and the SDA line is forced low by another device, the ARB flag (SPICR.5) is set and the SDO buffer is disabled (ARB is reset and the SDO buffer is enabled when SPIDR is written to again). When BMS is set, the peripheral clock is supplied through the INT2 line by the external clock line (SCL). Due to potential noise spikes (which must last longer than one INTCLK period to be detected), RX or TX may gain a clock pulse. Referring to Figure 67, if device ST9-1 detects a noise spike and therefore gains a clock pulse, it will stop its transmission early and hold the clock line low, causing device ST9-2 to freeze on the 7th bit. To exit and recover from this condition, the BMS bit must be reset; this will cause the SPI logic to be reset, thus aborting the current transmission. An End of Transmission interrupt is generated following this reset sequence.

Figure 67. SPI Arbitration



SERIAL PERIPHERAL INTERFACE (Cont'd)

9.4.7 S-Bus Interface

The S-bus is a three-wire bidirectional data-bus, possessing functional features similar to the I^2C-bus. As opposed to the I^2C-bus, the Start/Stop conditions are determined by encoding the information on 3 wires rather than on 2, as shown in Figure 69. The additional line is referred as SEN.

Figure 68. Mixed S-bus and I^2C-bus System

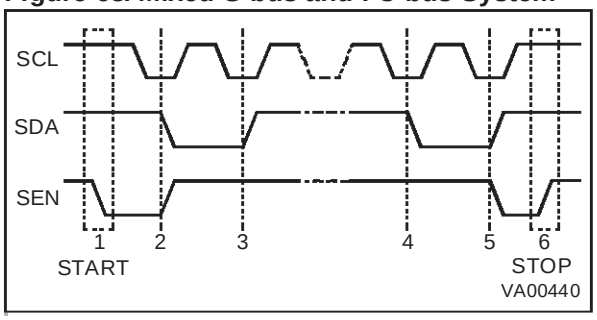
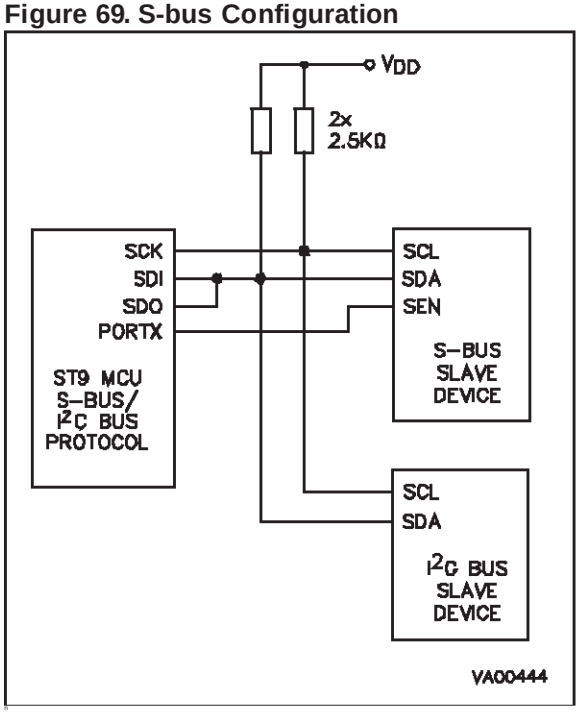


Figure 69. S-bus Configuration



SPI Working with S-bus

The S-bus protocol uses the same pin configuration as the I^2C-bus for generating the SCL and SDA lines. The additional SEN line is managed through a standard ST9 I/O port line, under software control (see Figure 65).

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.4.8 IM-bus Interface

The IM-bus features a bidirectional data line and a clock line; in addition, it requires an IDENT line to distinguish an address byte from a data byte (Figure 71). Unlike the I^2C-bus protocol, the IM-bus protocol sends the least significant bit first; this requires a software routine which reverses the bit order before sending, and after receiving, a data byte. Figure 70 shows the connections between an IM-bus peripheral and an ST9 SPI. The SDO and SDI pins are connected to the bidirectional data pin of the peripheral device. The SDO alternate function is configured as Open-Drain (external 2.5KΩ pull-up resistors are required).

With this type of configuration, data is sent to the peripheral by writing the data byte to the SPIDR register. To receive data from the peripheral, the user should write FFh to the SPIDR register, in order to generate the shift clock pulses. As the SDO line is set to the Open-Drain configuration, the incoming data bits that are set to "1" do not affect the SDO/SDI line status (which defaults to a high level due to the FFh value in the transmit register), while incoming bits that are set to "0" pull the input line low.

In software it is necessary to initialise the ST9 SPI by setting both CPOL and CPHA to "1". By using a general purpose I/O as the IDENT line, and forcing it to a logical "0" when writing to the SPIDR register, an address is sent (or read). Then, by setting this bit to "1" and writing to SPIDR, data is sent to the peripheral. When all the address and data pairs are sent, it is necessary to drive the IDENT line low and high to create a short pulse. This will generate the stop condition.

Figure 70. ST9 and IM-bus Peripheral

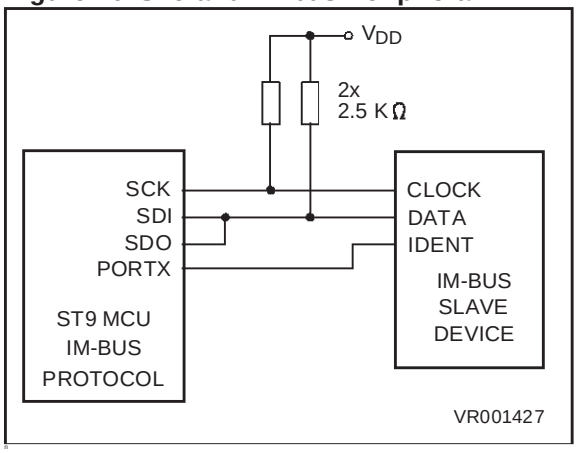
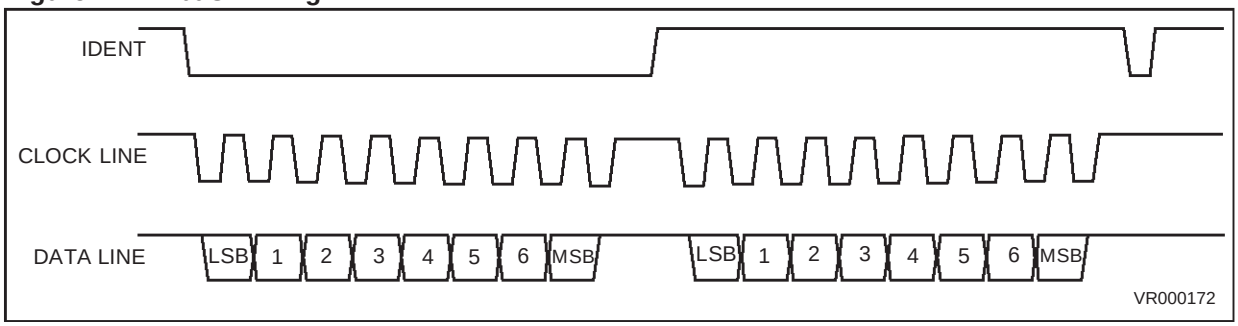


Figure 71. IM bus Timing



SERIAL PERIPHERAL INTERFACE (Cont'd)

9.4.9 Register Description

It is possible to have up to 3 independent SPIs in the same device (refer to the device block diagram). In this case they are named SPI0 thru SPI2. If the device has one SPI converter it uses the register adresses of SPI0. The register map is the following:

Register	SPIn	Page
SPIDR R253	SPI0	0
SPICR R254	SPI0	0
SPIDR1 R253	SPI1	7
SPICR1 R254	SPI1	7
SPIDR2 R245	SPI2	7
SPICR2 R246	SPI2	7

Note: In the register description on the following pages, register and page numbers are given using the example of SPI0.

SPI DATA REGISTER (SPIDR)
R253 - Read/Write
Register Page: 0
Reset Value: undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[0:7]**: *SPI Data*.
This register contains the data transmitted and received by the SPI. Data is transmitted bit 7 first, and incoming data is received into bit 0. Transmission is started by writing to this register.

Note: SPIDR state remains undefined until the end of transmission of the first byte.

SPI CONTROL REGISTER (SPICR)
R254 - Read/Write
Register Page: 0
Reset Value: 0000 0000 (00h)

7							0
SPEN	BMS	ARB	BUSY	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPEN**: *Serial Peripheral Enable*.
0: SCK and SDO are kept tristate.
1: Both alternate functions SCK and SDO are enabled.

Note: furthermore, SPEN (together with the BMS bit) affects the selection of the source for interrupt channel B0. Transmission starts when data is written to the SPIDR Register.

Bit 6 = **BMS**: *S-bus/I²C-bus Mode Selector*.
0: Perform a re-initialisation of the SPI logic, thus allowing recovery procedures after a RX/TX failure.
1: Enable S-bus/I²C-bus arbitration, clock synchronization and Start/ Stop detection (SPI used in an S-bus/I²C-bus protocol).

Note: when the BMS bit is reset, it affects (together with the SPEN bit) the selection of the source for interrupt channel B0.

Bit 5 = **ARB**: *Arbitration flag bit.*
This bit is set by hardware and can be reset by software.
0: S-bus/I²C-bus stop condition is detected.
1: Arbitration lost by the SPI in S-bus/I²C-bus mode.

Note: when ARB is set automatically, the SDO pin is set to a high value until a write instruction on SPIDR is performed.

Bit 4 = **BUSY**: *SPI Busy Flag*.
This bit is set by hardware. It allows the user to monitor the SPI status by polling its value.
0: No transmission in progress.
1: Transmission in progress.

Bit 3 = **CPOL**: *Transmission Clock Polarity*.
CPOL controls the normal or steady state value of the clock when data is *not* being transferred. Please refer to the following table and toFigure 72 to see this bit action (together with the CPHA bit).

Note: As the SCK line is held in a high impedance state when the SPI is disabled (SPEN = "0"), the SCK pin must be connected to V_{SS} or to V_{CC} through a resistor, depending on the CPOL state. Polarity should be set during the initialisation routine, in accordance with the setting of all peripherals, and should not be changed during program execution.

SERIAL PERIPHERAL INTERFACE (Cont'd)

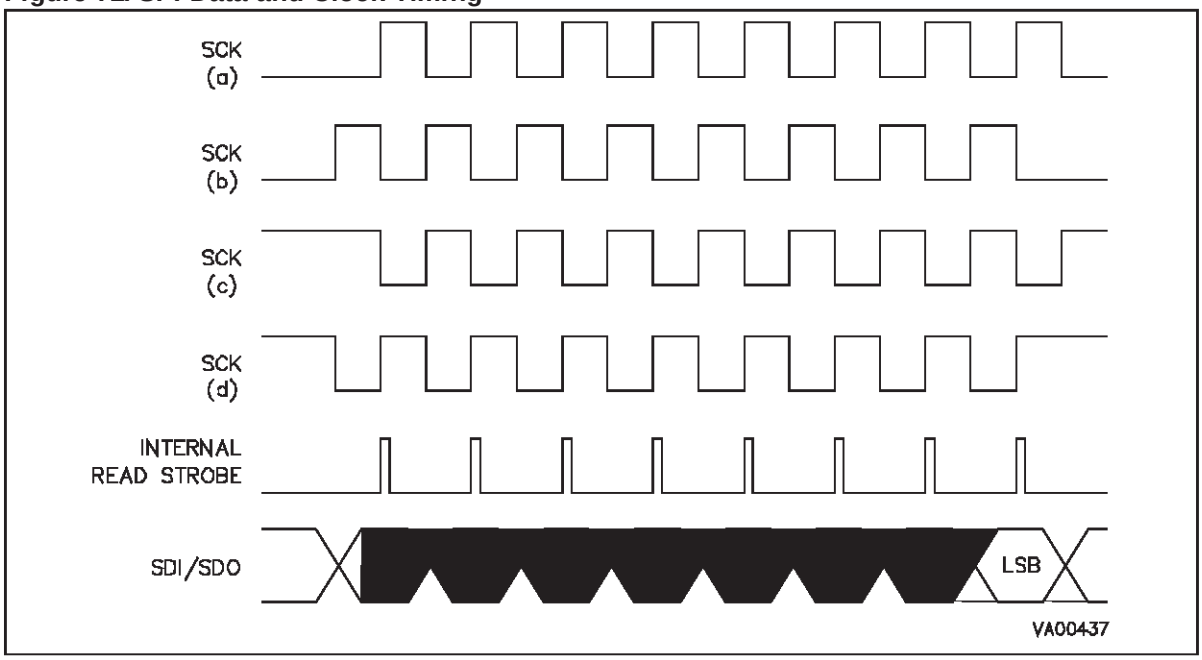
Bit 2 = **CPHA**: *Transmission Clock Phase.*
CPHA controls the relationship between the data on the SDI and SDO pins, and the clock signal on the SCK pin. The CPHA bit selects the clock edge used to capture data. It has its greatest impact on the first bit transmitted (MSB), because it does (or does not) allow a clock transition before the first data capture edge. Figure 72 shows the relationship between CPHA, CPOL and SCK, and indicates active clock edges and strobe times.

CPOL	CPHA	SCK (in Figure 72)
0	0	(a)
0	1	(b)
1	0	(c)
1	1	(d)

Bit 1:0 = **SPR[1:0]:** *SPI Rate.*
These two bits select one (of four) baud rates, to be used as SCK.

SPR1	SPR0	Clock Divider	SCK Frequency (@ INTCLK = 12MHz)	
0	0	8	1500kHz	(T = 0.67µs)
0	1	16	750kHz	(T = 1.33µs)
1	0	128	93.75kHz	(T = 10.66µs)
1	1	256	46.87kHz	(T = 21.33µs)

Figure 72. SPI Data and Clock Timing



9.5 SERIAL COMMUNICATIONS INTERFACE (SCI)

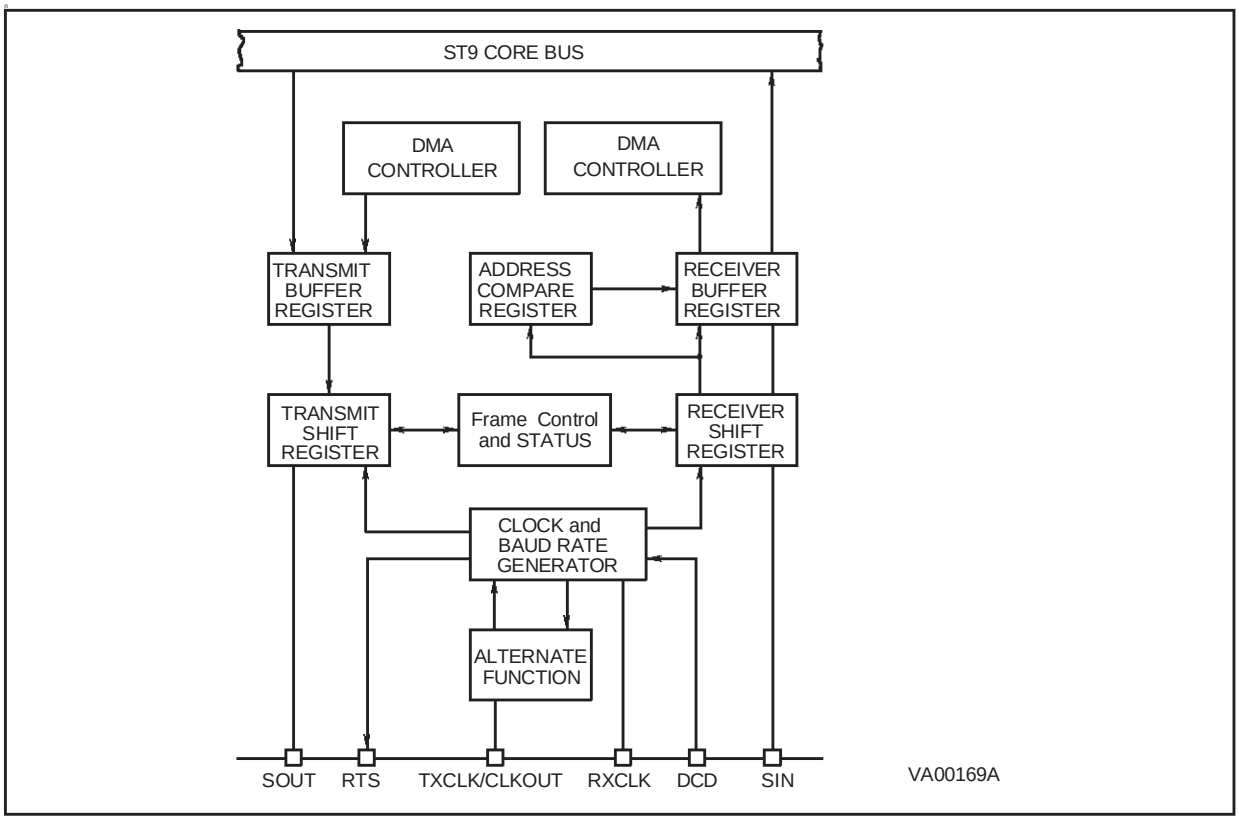
9.5.1 Introduction

The Serial Communications Interface (SCI) offers full-duplex serial data exchange with a wide range of external equipment. The SCI offers four operating modes: Asynchronous, Asynchronous with synchronous clock, Serial expansion and Synchronous.

The SCI offers the following principal features:

- Full duplex synchronous and asynchronous operation.
- Transmit, receive, line status, and device address interrupt generation.
- Integral Baud Rate Generator capable of dividing the input clock by any value from 2 to 2^{16}-1 (16 bit word) and generating the internal 16 X data sampling clock for asynchronous operation or the 1X clock for synchronous operation.
- Fully programmable serial interface:
  - 5, 6, 7, or 8 bit word length.
  - Even, odd, or no parity generation and detection.
  - 0, 1, 1-1/2, 2, 2-1/2, 3 stop bit generation.
  - Complete status reporting capabilities.
  - Line break generation and detection.
- Programmable address indication bit (wake-up bit) and user invisible compare logic to support multiple microcomputer networking. Optional character search function.
- Internal diagnostic capabilities:
  - Local loopback for communications link fault isolation.
  - Auto-echo for communications link fault isolation.
- Separate interrupt/DMA channels for transmit and receive.
- In addition, a Synchronous mode supports:
  - High speed communication
  - Possibility of hardware synchronization (RTS/DCD signals).
  - Programmable polarity and stand-by level for data SIN/SOUT.
  - Programmable active edge and stand-by level for clocks CLKOUT/RXCL.
  - Programmable active levels of RTS/DCD signals.
  - Full Loop-Back and Auto-Echo modes for DATA, CLOCKs and CONTROLs.

Figure 73. SCI Block Diagram



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

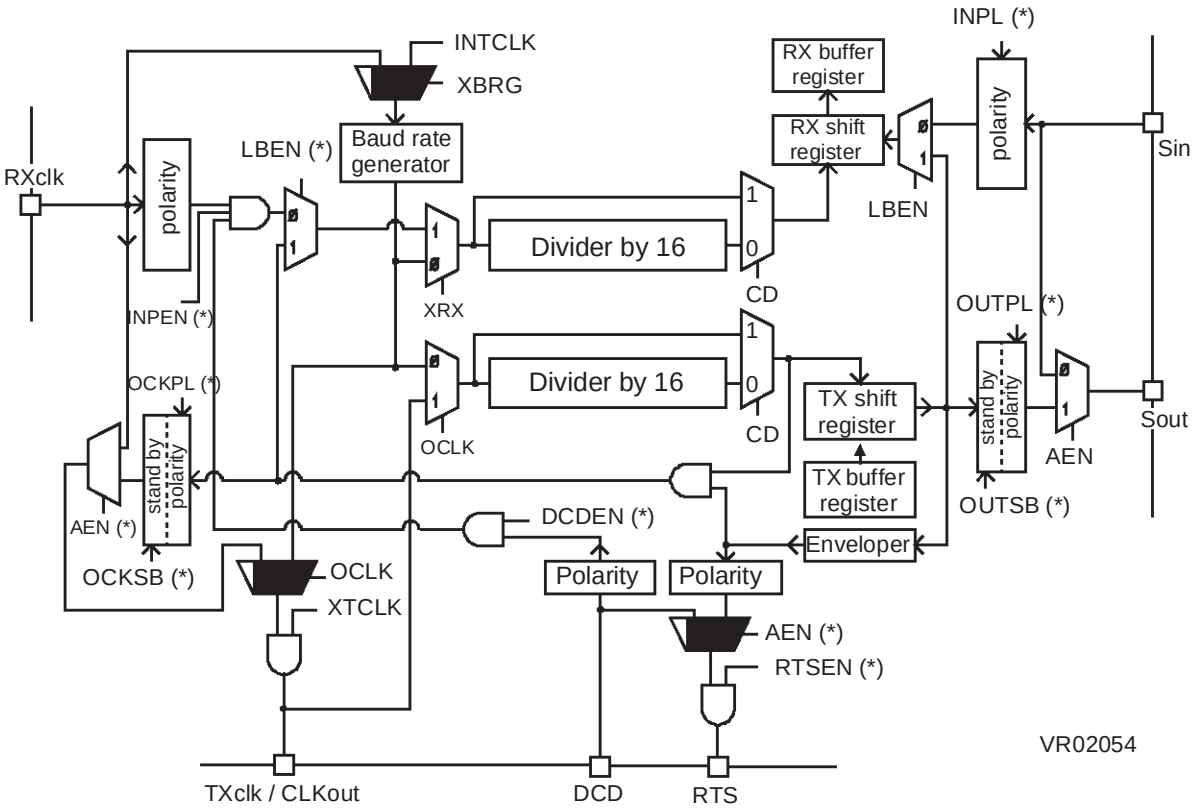
9.5.2 Functional Description

The SCI offers four operating modes:

– Asynchronous mode
– Asynchronous mode with synchronous clock
– Serial expansion mode
– Synchronous mode

Asynchronous mode, Asynchronous mode with synchronous clock and Serial expansion mode output data with the same serial frame format. The differences lie in the data sampling clock rates (1X, 16X) and in the operating modes.

Figure 74. SCI Functional Schematic



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.3 SCI Operating Modes

9.5.3.1 Asynchronous Mode

In this mode, data and clock can be asynchronous (the transmitter and receiver can use their own clocks to sample received data), each data bit is sampled 16 times per clock period.

The baud rate clock should be set to the ÷16 Mode and the frequency of the input clock (from an external source or from the internal baud-rate generator output) is set to suit.

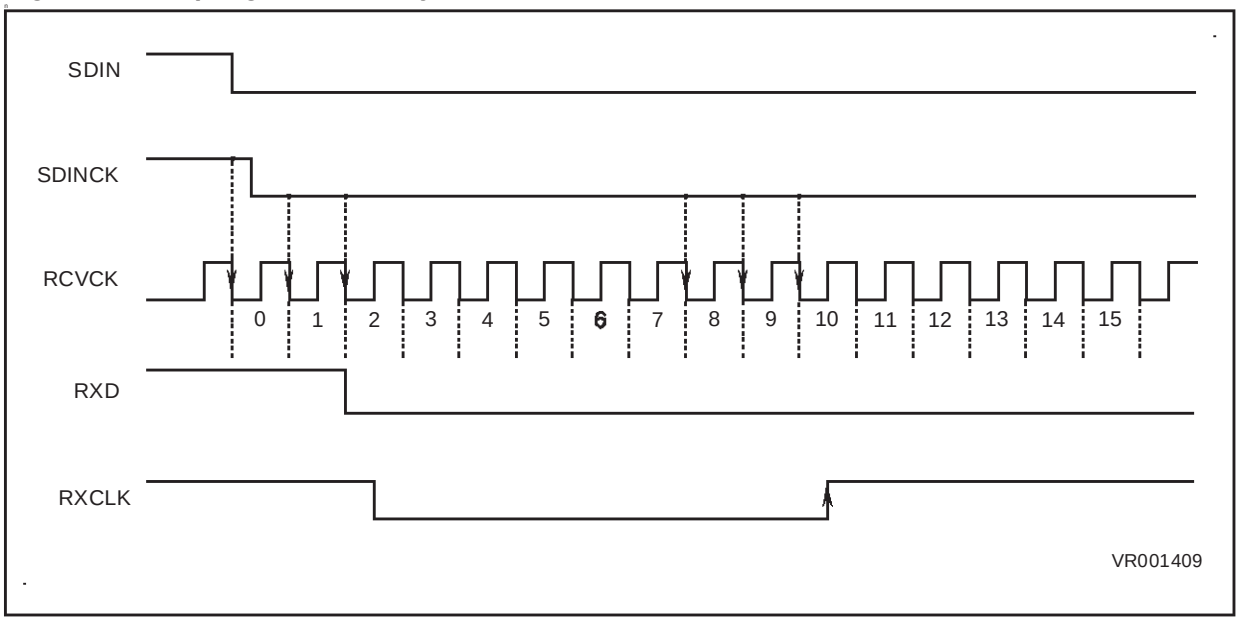
9.5.3.2 Asynchronous Mode with Synchronous Clock

In this mode, data and clock are synchronous, each data bit is sampled once per clock period.

For transmit operation, a general purpose I/O port pin can be programmed to output the CLKOUT signal from the baud rate generator. If the SCI is provided with an external transmission clock source, there will be a skew equivalent to two INTCLK periods between clock and data.

Data will be transmitted on the falling edge of the transmit clock. Received data will be latched into the SCI on the rising edge of the receive clock.

Figure 75. Sampling Times in Asynchronous Format



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.3.3 Serial Expansion Mode

This mode is used to communicate with an external synchronous peripheral.

The transmitter only provides the clock waveform during the period that data is being transmitted on the CLKOUT pin (the Data Envelope). Data is latched on the rising edge of this clock.

Whenever the SCI is to receive data in serial port expansion mode, the clock must be supplied externally, and be synchronous with the transmitted data. The SCI latches the incoming data on the rising edge of the received clock, which is input on the RXCLK pin.

9.5.3.4 Synchronous Mode

This mode is used to access an external synchronous peripheral, dummy start/stops bits are not included in the data frame. Polarity, stand-by level and active edges of I/O signals are fully and separately programmable for both inputs and outputs.

It's necessary to set the SMEN bit of the Synchronous Input Control Register (SICR) to enable this mode and all the related extra features (otherwise disabled).

The transmitter will provide the clock waveform only during the period when the data is being transmitted via the CLKOUT pin, which can be enabled by setting both the XTCLK and OCLK bits of the Clock Configuration Register. Whenever the SCI is to receive data in synchronous mode, the clock waveform must be supplied externally via the RXCLK pin and be synchronous with the data. For correct receiver operation, the XRX bit of the Clock Configuration Register must be set.

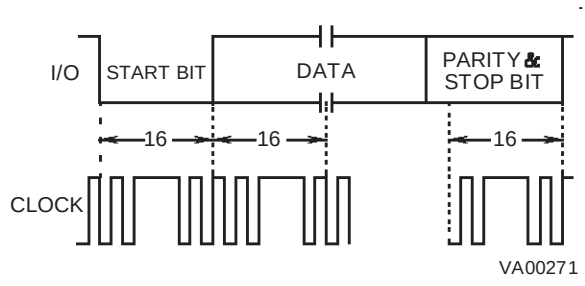
Two external signals, Request-To-Send and Data-Carrier-Detect (RTS/DCD), can be enabled to synchronise the data exchange between two serial units. The RTS output becomes active just before the first active edge of CLKOUT and indicates to the target device that the MCU is about to send a synchronous frame; it returns to its stand-by state following the last active edge of CLKOUT (MSB transmitted).

The DCD input can be considered as a gate that filters RXCLK and informs the MCU that a transmitting device is transmitting a data frame. Polarity of RTS/DCD is individually programmable, as for clocks and data.

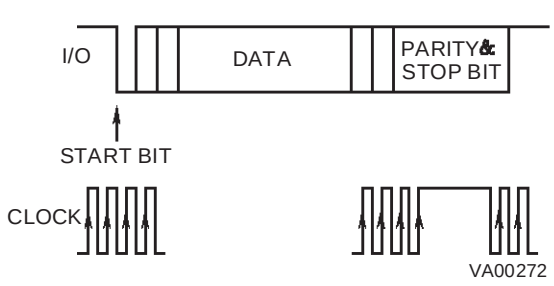
The data word is programmable from 5 to 8 bits, as for the other modes; parity, address/9th, stop bits and break cannot be inserted into the transmitted frame. Programming of the related bits of the SCI control registers is irrelevant in Synchronous Mode: all the corresponding interrupt requests must, in any case, be masked in order to avoid incorrect operation during data reception.

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

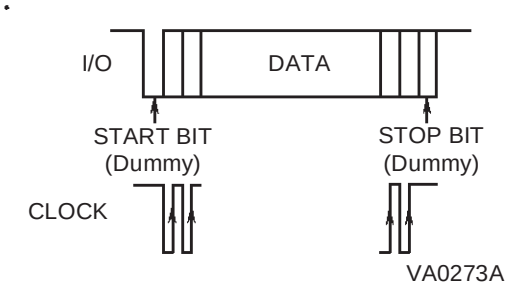
Figure 76. SCI Operating Modes



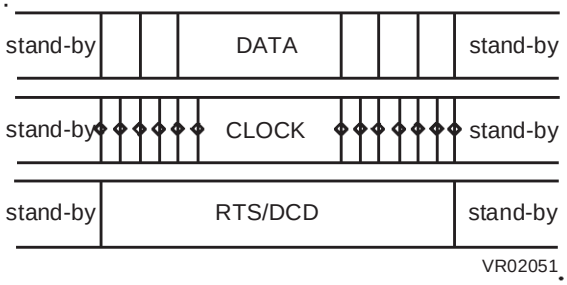
Asynchronous Mode



Asynchronous Mode with Synchronous Clock



Serial Expansion Mode



Synchronous Mode

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.4 Serial Frame Format

Characters sent or received by the SCI can have some or all of the features in the following format, depending on the operating mode:

START: the START bit indicates the beginning of a data frame in Asynchronous modes. The START condition is detected as a high to low transition.
A dummy START bit is generated in Serial Expansion mode.

DATA: the DATA word length is programmable from 5 to 8 bits, for both Synchronous and Asynchronous modes. LSB are transmitted first.

PARITY: The Parity Bit (not available in Serial Expansion mode and Synchronous mode) is optional, and can be used with any word length. It is used for error checking and is set so as to make the total number of high bits in DATA plus PARITY odd or even, depending on the number of "1"s in the DATA field.

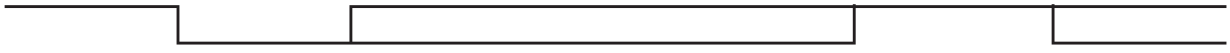
ADDRESS/9TH: The Address/9th Bit is optional and may be added to any word format. It is used in both Serial Expansion and Asynchronous modes to indicate that the data is an address (bit set).

The ADDRESS/9TH bit is useful when several microcontrollers are exchanging data on the same serial bus. Individual microcontrollers can stay idle on the serial bus, waiting for a transmitted address. When a microcontroller recognizes its own address, it can begin Data Reception, likewise, on the transmit side, the microcontroller can transmit another address to begin communication with a different microcontroller.

The ADDRESS/9TH bit can be used as an additional data bit or to mark control words (9th bit).

STOP: Indicates the end of a data frame in Asynchronous modes. A dummy STOP bit is generated in Serial Expansion mode. The STOP bit can be programmed to be 0, 1, 1.5, 2, 2.5 or 3 bits long, depending on the mode. It returns the SCI to the quiescent marking state (i.e., a constant high-state condition) which lasts until a new start bit indicates an incoming word.

Figure 77. SCI Character Formats



	START(2)	DATA(1)	PARITY(3)	ADDRESS(2)	STOP(2)	
# bits	0, 1	5, 6, 7, 8	0, 1	0, 1	0, 1, 1.5, 2, 2.5, 1, 2, 3	16X 1X
states			NONE ODD EVEN	ON OFF		

(1) LSB First

(2) Not available in Synchronous mode

(3) Not available in Serial Expansion mode and Synchronous mode

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.4.1 Data transfer

Data to be transmitted by the SCI is first loaded by the program into the Transmitter Buffer Register. The SCI will transfer the data into the Transmitter Shift Register when the Shift Register becomes available (empty). The Transmitter Shift Register converts the parallel data into serial format for transmission via the SCI Alternate Function output, Serial Data Out. On completion of the transfer, the transmitter buffer register interrupt pending bit will be updated. If the selected word length is less than 8 bits, the unused most significant bits do not need to be defined.

Incoming serial data from the Serial Data Input pin is converted into parallel format by the Receiver Shift Register. At the end of the input data frame, the valid data portion of the received word is transferred from the Receiver Shift Register into the Receiver Buffer Register. All Receiver interrupt conditions are updated at the time of transfer. If the selected character format is less than 8 bits, the unused most significant bits will be set.

The Frame Control and Status block creates and checks the character configuration (Data length and number of Stop bits), as well as the source of the transmitter/receiver clock.

The internal Baud Rate Generator contains a programmable divide by "N" counter which can be used to generate the clocks for the transmitter and/or receiver. The baud rate generator can use INTCLK or the Receiver clock input via RXCLK.

The Address bit/D9 is optional and may be added to any word in Asynchronous and Serial Expansion modes. It is commonly used in network or machine control applications. When enabled (AB set), an address or ninth data bit can be added to a transmitted word by setting the Set Address bit (SA). This is then appended to the next word entered into the (empty) Transmitter Buffer Register and then cleared by hardware. On character input, a set Address Bit can indicate that the data preceding the bit is an address which may be compared in hardware with the value in the Address Compare Register (ACR) to generate an Address Match interrupt when equal.

The Address bit and Address Comparison Register can also be combined to generate four different types of Address Interrupt to suit different protocols, based on the status of the Address Mode Enable bit (AMEN) and the Address Mode bit (AM) in the CHCR register.

The character match Address Interrupt mode may be used as a powerful character search mode, generating an interrupt on reception of a predetermined character e.g. Carriage Return or End of Block codes (Character Match Interrupt). This is the only Address Interrupt Mode available in Synchronous mode.

The Line Break condition is fully supported for both transmission and reception. Line Break is sent by setting the SB bit (IDPR). This causes the transmitter output to be held low (after all buffered data has been transmitted) for a minimum of one complete word length and until the SB bit is Reset. Break cannot be inserted into the transmitted frame for the Synchronous mode.

Testing of the communications channel may be performed using the built-in facilities of the SCI peripheral. Auto-Echo mode and Loop-Back mode may be used individually or together. In Asynchronous, Asynchronous with Synchronous Clock and Serial Expansion modes they are available only on SIN/SOUT pins through the programming of AEN/LBEN bits in CCR. In Synchronous mode (SMEN set) the above configurations are available on SIN/SOUT, RXCLK/CLKOUT and DCD/RTS pins by programming the AEN/LBEN bits and independently of the programmed polarity. In the Synchronous mode case, when AEN is set, the transmitter outputs (data, clock and control) are disconnected from the I/O pins, which are driven directly by the receiver input pins (Auto-Echo mode: SOUT=SIN, CLKOUT=RXCLK and RTS=DCD, even if they act on the internal receiver with the programmed polarity/edge). When LBEN is set, the receiver inputs (data, clock and controls) are disconnected and the transmitter outputs are looped-back into the receiver section (Loop-Back mode: SIN=SOUT, RXCLK=CLKOUT, DCD=RTS. The output pins are locked to their programmed stand-by level and the status of the INPL, XCKPL, DCDPL, OUTPL, OCKPL and RTSPL bits in the SICR register are irrelevant). Refer to Figure 78, Figure 79, and Figure 80 for these different configurations.

Table 29. Address Interrupt Modes

If 9th Data Bit is set (1)
If Character Match
If Character Match and 9th Data Bit is set(1)
If Character Match Immediately Follows BREAK (1)

(1) Not available in Synchronous mode

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

Figure 78. Auto Echo Configuration

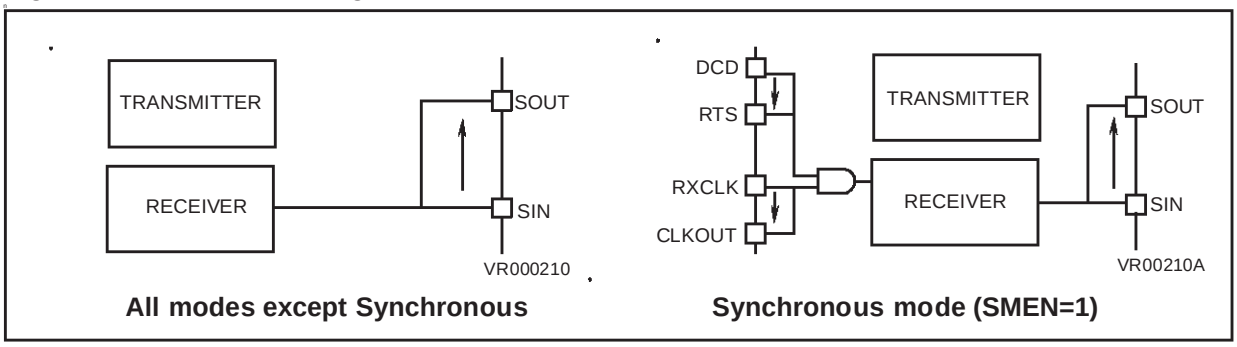


Figure 79. Loop Back Configuration

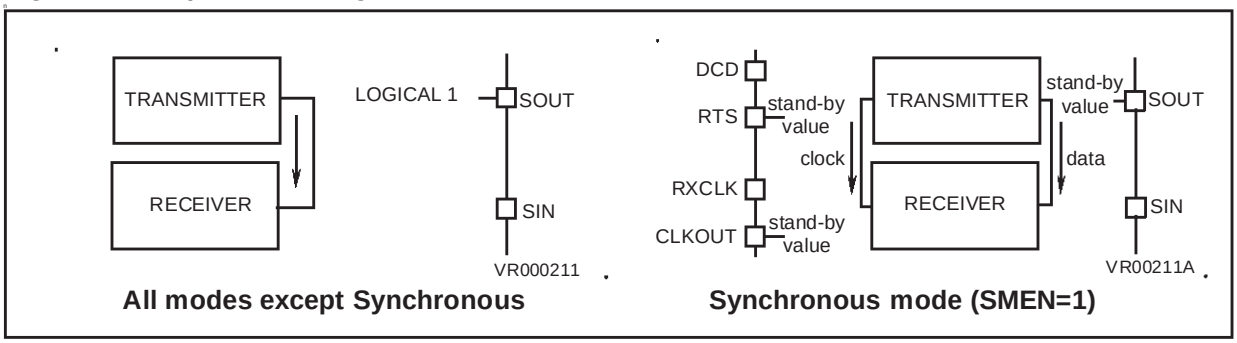
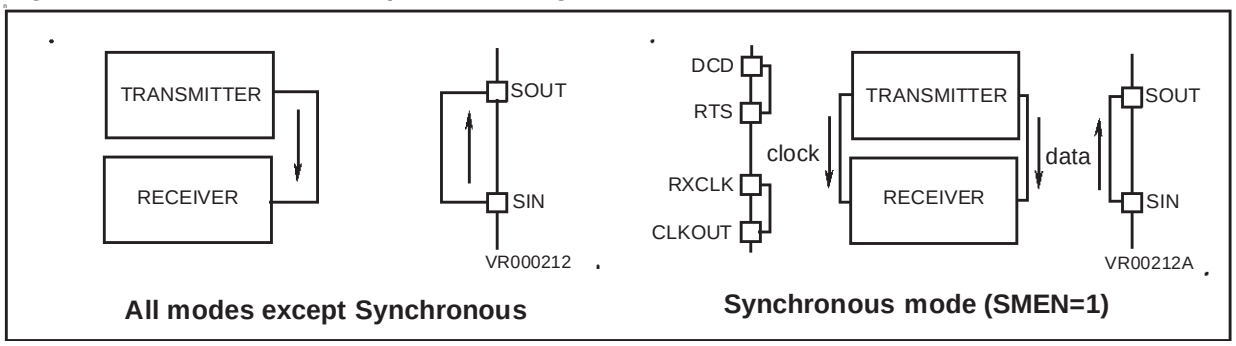


Figure 80. Auto Echo and Loop-Back Configuration



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.5 Clocks And Serial Transmission Rates

The communication bit rate of the SCI transmitter and receiver sections can be provided from the internal Baud Rate Generator or from external sources. The bit rate clock is divided by 16 in Asynchronous mode (CD in CCR reset), or undivided in the 3 other modes (CD set). With INTCLK running at 20MHz, or with a 10MHz external clock, a maximum bit rate of 5MBaud is available in undivided mode and 625KBaud or 312.5KBaud respectively in divided by 16 mode.

External Clock Sources. The External Clock input pin TXCLK may be programmed by the XTCLK and OCLK bits in the CCR register as: the transmit clock input, Baud Rate Generator output (allowing an external divider circuit to provide the receive clock for split rate transmit and receive), or as CLKOUT output in Synchronous and Serial Expansion modes. The RXCLK Receive clock input is enabled by the XRX bit, this input should be set in accordance with the setting of the CD bit.

Baud Rate Generator. The internal Baud Rate Generator consists of a 16-bit programmable divide by "N" counter which can be used to generate the transmitter and/or receiver clocks. The minimum baud rate divisor is 2 and the maximum divisor is 2^{16}-1. After initialising the baud rate generator, the divisor value is immediately loaded into the counter. This prevents potentially long random counts on the initial load.
The Baud Rate generator frequency is equal to the Input Clock frequency divided by the Divisor value.

WARNING: *Programming the baud rate divider to 0 or 1 will stop the divider.*

The output of the Baud Rate generator has a precise 50% duty cycle. The Baud Rate generator can use INTCLK for the input clock source. In this case, INTCLK (and therefore the MCU Xtal) should be chosen to provide a suitable frequency for division by the Baud Rate Generator to give the required transmit and receive bit rates. Suitable INTCLK frequencies and the respective divider values for standard Baud rates are shown inTable 30 Practical Example of SCI Baud Rate Generator Divider Values.

9.5.6 SCI Initialization Procedure

Writing to either of the two Baud Rate Generator Registers immediately disables and resets the SCI baud rate generator, as well as the transmitter and receiver circuitry.

After writing to the second Baud Rate Generator Register, the transmitter and receiver circuits are enabled. The Baud Rate Generator will load the new value and start counting.

To initialize the SCI, the user should first initialize the most significant byte of the Baud Rate Generator Register; this will reset all SCI circuitry. The user should then initialize all other SCI registers (SICR/SOCR included) for the desired operating mode and then, to enable the SCI, he should initialize the least significant byte Baud Rate Generator Register.

'On-the-Fly' modifications of the control registers' content during transmitter/receiver operations, although possible, can corrupt data and produce undesirable spikes on the I/O lines (data, clock and control). Furthermore, modifying the control registers' content without reinitialising the SCI circuitry (during stand-by cycles, waiting to transmit or receive data) must be kept carefully under control by software to avoid spurious data being transmitted or received.

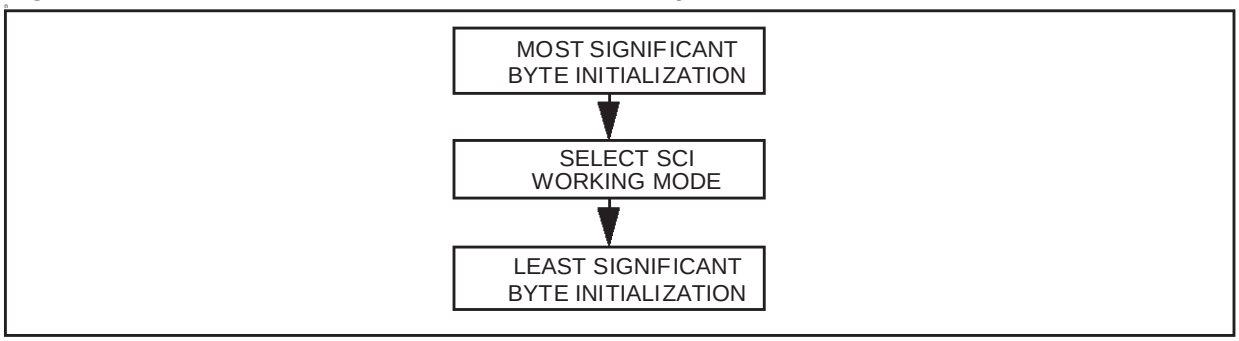
Note: For synchronous receive operation, the data and receive clock must not exhibit significant skew between clock and data. The received data and clock are internally synchronized to INTCLK.

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

Table 30. Practical Example of SCI Baud Rate Generator Divider Values

INTCLK: 19660.800 KHz							
Baud Rate	Clock Factor	Desired Freq (kHz)	Divisor		Actual Baud Rate	Actual Freq (kHz)	Deviation
			Dec	Hex			
50.00	16 X	0.80000	24576	6000	50.00	0.80000	0.0000%
75.00	16 X	1.20000	16384	4000	75.00	1.20000	0.0000%
110.00	16 X	1.76000	11170	2BA2	110.01	1.76014	-0.00081%
300.00	16 X	4.80000	4096	1000	300.00	4.80000	0.0000%
600.00	16 X	9.60000	2048	800	600.00	9.60000	0.0000%
1200.00	16 X	19.20000	1024	400	1200.00	19.20000	0.0000%
2400.00	16 X	38.40000	512	200	2400.00	38.40000	0.0000%
4800.00	16 X	76.80000	256	100	4800.00	76.80000	0.0000%
9600.00	16 X	153.60000	128	80	9600.00	153.60000	0.0000%
19200.00	16 X	307.20000	64	40	19200.00	307.20000	0.0000%
38400.00	16 X	614.40000	32	20	38400.00	614.40000	0.0000%
76800.00	16 X	1228.80000	16	10	76800.00	1228.80000	0.0000%

Figure 81. SCI Baud Rate Generator Initialization Sequence



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.7 Input Signals

SIN: Serial Data Input This pin is the serial data input to the SCI receiver shift register.

TXCLK: External Transmitter Clock Input This pin is the external input clock driving the SCI transmitter. The TXCLK frequency must be greater than or equal to 16 times the transmitter data rate (depending whether the X16 or the X1 clock have been selected) and must have a period of at least twice INTCLK. The use of the TXCLK pin is optional.

RXCLK: External Receiver Clock Input. This input is the clock to the SCI receiver when using an external clock source connected to the baud rate generator. INTCLK is normally the clock source. A 50% duty cycle is not required for this input, however, the shortest period must last more than two INTCLK periods. Use of RXCLK is optional.

DCD: Data Carrier Detect. This input is enabled only in Synchronous mode; it works as a gate for the RXCLK clock and informs the MCU that an emitting device is transmitting a synchronous frame. The active level can be programmed as 1 or 0 and must be provided at least one INTCLK period before the first active edge of the input clock.

9.5.8 Output Signals

SOUT: Serial Data Output. This Alternate Function output signal is the serial data output for the SCI transmitter in all operating modes.

CLKOUT: Clock Output The alternate Function of this pin outputs either the data clock from the transmitter in Serial Expansion or Synchronous modes, or the clock output from the Baud Rate Generator. In Serial expansion mode it will clock only the data portion of the frame and its stand-by state is high: data is valid on the rising edge of the clock. Even in Synchronous mode CLKOUT will only clock the data portion of the frame, but the stand-by level and active edge polarity are programmable by the user.

When Synchronous mode is disabled (SMEN in SICR is reset), the state of the XTCLK and OCLK bits in CCR determine the source of CLKOUT; '11' enables the Serial Expansion Mode.

When the Synchronous mode is enabled (SMEN in SICR is set), the state of the XTCLK and OCLK bits in CCR determine the source of CLKOUT; '00' disables it for PLM applications.

RTS: Request To Send. This output Alternate Function is only enabled in Synchronous mode; it becomes active when the Least Significant Bit of the data frame is sent to the Serial Output Pin (SOUT) and indicates to the target device that the MCU is about to send a synchronous frame; it returns to its stand-by value just after the last active edge of CLKOUT (MSB transmitted). The active level can be programmed high or low.

9.5.9 Interrupts and DMA

9.5.9.1 Interrupts

The SCI can generate interrupts as a result of several conditions. Receiver interrupts include data pending, receive errors (overrun, framing and parity), as well as address or break pending. Transmitter interrupts are software selectable for either Transmit Buffer Register Empty (BSN set) or for Transmit Shift Register Empty (BSN reset) conditions.

Typical usage of the Interrupts generated by the SCI peripheral are illustrated inFigure 82.

The SCI peripheral is able to generate interrupt requests as a result of a number of events, several of which share the same interrupt vector. It is therefore necessary to poll ISR, the Interrupt Status Register, in order to determine the active trigger. These bits should be reset by the programmer during the Interrupt Service routine.

The four major levels of interrupt are encoded in hardware to provide two bits of the interrupt vector register, allowing the position of the block of pointer vectors to be resolved to an 8 byte block size.

The SCI interrupts have an internal priority structure in order to resolve simultaneous events. Refer also to Section 9.5.3 SCI Operating Modes for more details relating to Synchronous mode.

Table 31. SCI Interrupt Internal Priority

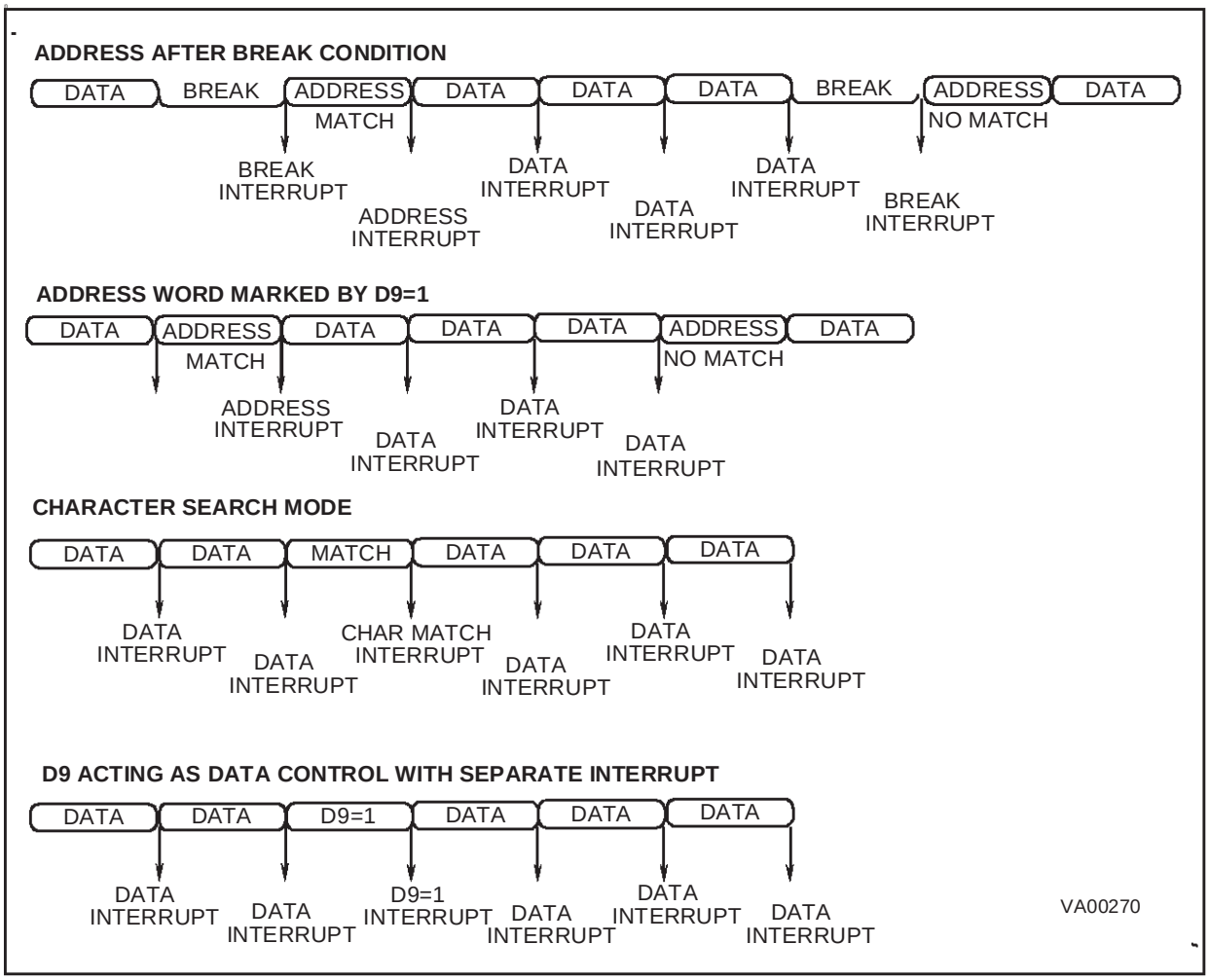
Receive DMA Request	Highest Priority
Transmit DMA Request	
Receive Interrupt	
Transmit Interrupt	Lowest Priority

SERIAL COMMUNICATIONS INTERFACE(Cont'd)

Table 32. SCI Interrupt Vectors

Interrupt Source	Vector Address
Transmitter Buffer or Shift Register Empty Transmit DMA end of Block	xxx x110
Received Ready Receive DMA end of Block	xxxx x100
Break Detector Address Word Match	xxxx x010
Receiver Error	xxxx x000

Figure 82. SCI Interrupts: Example of Typical Usage



SERIAL COMMUNICATIONS INTERFACE(Cont'd)

9.5.9.2 DMA

Two DMA channels are associated with the SCI, for transmit and for receive. These follow the register scheme as described in DMA chapter. It should be noted that, after initializing the DMA counter and pointer registers and enabling DMA, data transmission is triggered by a character written into the Transmit Buffer register.

When DMA is active the Receive Data Pending bit (RXDP in ISR), and the Transmit status bit interrupt sources are replaced by the DMA End Of Block Interrupt sources for transmit and receive, respectively.

The last DMA data word of a block of data will cause a DMA cycle followed by a transmit interrupt. This sequence will signal to the ST9 core to reinitialize the transmit DMA block counter. The Transmit End of Block status bit (TXEOB) should be reset by software in order to avoid undesired interrupt routines, especially in the initialisation routine (after reset) and after entering the End Of Block interrupt routine.

Similarly the last DMA data word of a block of data will cause a DMA cycle followed by a receiver data ready interrupt. This sequence will signal to the ST9 core to reinitialize the receiver DMA block counter. The Received End of Block status bit (RXEOB) should be reset by software in order to avoid undesired interrupt routines, especially in the initialisation routine (after reset) and after entering the End Of Block interrupt routine.

Remark: If properly initialized, the DMA controller starts a data transfer after and only if the running program has loaded the Transmitter Buffer Register with a value. In order to execute properly a DMA transmission, the End Of Block interrupt routine must include the following actions:

– Load the Transmitter Buffer Register (TXBR) with the first byte to transmit.
– Restore the DMA counter (TDCPR)
– Restore the DMA pointer (TDAPR)
– Reset the transmitter end of block bit TXEOB (IMR.5)
– Reset the transmitter Buffer empty bit TXBEM (ISR.1)
– Enable DMA

9.5.10 Register Description

The SCI registers are located in the following pages in the ST9:

SCI number 0: page 24 (18h)

SCI number 1: page 25 (19h) (when present)

The SCI is controlled by the following registers:

Address	Register
R240 (F0h)	Receiver DMA Transaction Counter Pointer Register
R241 (F1h)	Receiver DMA Source Address Pointer Register
R242 (F2h)	Transmitter DMA Transaction Counter Pointer Register
R243 (F3h)	Transmitter DMA Destination Address Pointer Register
R244 (F4h)	Interrupt Vector Register
R245 (F5h)	Address Compare Register
R246 (F6h)	Interrupt Mask Register
R247 (F7h)	Interrupt Status Register
R248 (F8h)	Receive Buffer Register same Address as Transmitter Buffer Register (Read Only)
R248 (F8h)	Transmitter Buffer Register same Address as Receive Buffer Register (Write only)
R249 (F9h)	Interrupt/DMA Priority Register
R250 (FAh)	Character Configuration Register
R251 (FBh)	Clock Configuration Register
R252 (FCh)	Baud Rate Generator High Register
R253 (FDh)	Baud Rate Generator Low Register
R254 (FEh)	Synchronous Input Control Register
R255 (FFh)	Synchronous Output Control Register

REGISTER DESCRIPTION (Cont'd)

RECEIVER DMA COUNTER POINTER (RDCPR)

R240 - Read/Write

Reset value: undefined

7							0
RC7	RC6	RC5	RC4	RC3	RC2	RC1	RR/M

Bit 7:1 = **RC[7:1]**: *Receiver DMA Counter Pointer.* These bits contain the address of the pointer (in the Register File) of the receiver DMA transaction counter.

Bit 0 = **RR/M**: *Receiver Register File/Memory Selector.*
0: Select Memory space as destination.
1: Select the Register File as destination.

RECEIVER DMA ADDRESS POINTER (RDAPR)

R241 - Read/Write

Reset value: undefined

7							0
RA7	RA6	RA5	RA4	RA3	RA2	RA1	RD/P

Bit 7:1 = **RA[7:1]**: *Receiver DMA Address Pointer.* These bits contain the address of the pointer (in the Register File) of the receiver DMA data source.

Bit 0 = **RPS**: *Receiver DMA Memory Pointer Selector.*
This bit is only significant if memory has been selected for DMA transfers (RR/M = 0 in the RDCPR register).
0: Select ISR register for receiver DMA transfers address extension.
1: Select DMASR register for receiver DMA transfers address extension.

TRANSMITTER DMA COUNTER POINTER (TDCPR)

R242 - Read/Write

Reset value: undefined

7							0
TC7	TC6	TC5	TC4	TC3	TC2	TC1	TR/M

Bit 7:1 = **TC[7:1]**: *Transmitter DMA Counter Pointer.*
These bits contain the address of the pointer (in the Register File) of the transmitter DMA transaction counter.

Bit 0 = **TR/M**: *Transmitter Register File/Memory Selector.*
0: Select Memory space as source.
1: Select the Register File as source.

TRANSMITTER DMA ADDRESS POINTER (TDAPR)

R243 - Read/Write

Reset value: undefined

7							0
TA7	TA6	TA5	TA4	TA3	TA2	TA1	TD/P

Bit 7:1 = **TA[7:1]**: *Transmitter DMA Address Pointer.*
These bits contain the address of the pointer (in the Register File) of the transmitter DMA data source.

Bit 0 = **TPS**: *Transmitter DMA Memory Pointer Selector.*
This bit is only significant if memory has been selected for DMA transfers (TR/M = 0 in the TDCPR register).
0: Select ISR register for transmitter DMA transfers address extension.
1: Select DMASR register for transmitter DMA transfers address extension.

REGISTER DESCRIPTION (Cont'd)

INTERRUPT VECTOR REGISTER (IVR)

R244 - Read/Write

Reset value: undefined

7							0
V7	V6	V5	V4	V3	EV2	EV1	0

Bit 7:3 = **V[7:3]**: *SCI Interrupt Vector Base Address.*
User programmable interrupt vector bits for transmitter and receiver.

Bit 2:1 = **EV[2:1]**: *Encoded Interrupt Source.*
Both bits EV2 and EV1 are read only and set by hardware according to the interrupt source.

EV2	EV1	Interrupt source
0	0	Receiver Error (Overrun, Framing, Parity)
0	1	Break detect or address match
1	0	Receiver data ready/receiver DMA End of Block
1	1	Transmitter buffer or shift register empty transmitter DMA End of Block

Bit 0 = **D0**: This bit is forced by hardware to 0.

ADDRESS/DATA COMPARE REGISTER (ACR)

R245 - Read/Write

Reset value: undefined

7							0
AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0

Bit 7:0 = **AC[7:0]**: *Address/Compare Character.*
With either 9th bit address mode, address after break mode, or character search, the received address will be compared to the value stored in this register. When a valid address matches this register content, the Receiver Address Pending bit (RXAP in the ISR register) is set. After the RXAP bit is set in an addressed mode, all received data words will be transferred to the Receiver Buffer Register.

INTERRUPT MASK REGISTER (IMR)

R246 - Read/Write

Reset value: 0xx00000

7							0
BSN	RXEOB	TXEOB	RXE	RXA	RXB	RXDI	TXDI

Bit 7 = **BSN**: *Buffer or shift register empty interrupt.*
This bit selects the source of the transmitter register empty interrupt.
0:Select a shift register empty as source of a transmitter register empty interrupt.
1: Select a Buffer register empty as source of a transmitter register empty interrupt.

Bit 6 = **RXEOB**: *Received End of Block.*
This bit is set by hardware only and should be reset by software in order to avoid undesired interrupt routines, especially in the initialisation routine (after reset) and after entering the End Of Block interrupt routine. RXEOB is set after a receiver DMA cycle to mark the end of a block of data. The last DMA data word will cause a DMA cycle followed by a receiver data ready interrupt. This sequence instructs the ST9 core to reinitialize the receiver DMA block counter.
0: Cancel the interrupt request.
1: Mark the end of a received block of data.

Bit 5 = **TXEOB**: *Transmitter End of Block.*
The same comments made for RXEOB apply. TXEOB is set in a transmitter DMA cycle to mark the end of a data block. The last DMA data word will cause a DMA cycle followed by a transmitter interrupt. This sequence instructs the ST9 core to reinitialize the transmitter DMA block counter.
0: Cancel the interrupt request.
1: Mark the end of a transmitted block of data.

Bit 4 = **RXE**: *Receiver Error Mask.*
0: Disable Receiver error interrupts (OE, PE, and FE pending bits in the ISR register).
1: Enable Receiver error interrupts.

Bit 3 = **RXA**: *Receiver Address Mask*
0: Disable Receiver Address interrupt (RXAP pending bit in the ISR register).
1: Enable Receiver Address interrupt.

REGISTER DESCRIPTION (Cont'd)

Bit 2 = **RXB**: *Receiver Break Mask*
0: Disable Receiver Break interrupt (RXBP pending bit in the ISR register).
1: Enable Receiver Break interrupt.

Bit 1 = **RXDI**: *Receiver Data Interrupt Mask*
0: Disable Receiver Data and Receiver End of Block interrupts (RXDP and RXEOB pending bits in the ISR register).
1: Enable Receiver Data and Receiver End of Block interrupts.

Note: RXDI has no effect on DMA transfers.

Bit 0 = **TXDI**: *Transmitter Data Interrupt Mask*
0: Disable Transmitter Buffer Register Empty, Transmitter Shift Register Empty, or Transmitter End of Block interrupts (TXBEM, TXSEM, and TXEOB bits in the ISR register).
1: Enable Transmitter Buffer Register Empty, Transmitter Shift Register Empty, or Transmitter End of Block interrupts.

Note: TXDI has no effect on DMA transfers.

INTERRUPT STATUS REGISTER (ISR)

R247 - Read/Write

Reset value: undefined

7							0
OE	FE	PE	RXAP	RXBP	RXDP	TXBEM	TXSEM

Bit 7 = **OE**: *Overrun Error Pending*.
This bit is set by hardware if the data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register (the previous data is lost).
0: No Overrun Error.
1: Overrun Error occurred.

Bit 6 = **FE**: *Framing Error Pending bit*
This bit is set by hardware if the received data word did not have a valid stop bit.
0: No Framing Error.
1: Framing Error occurred.

Note: In the case where a framing error occurs when the SCI is programmed in address mode and is monitoring an address, the interrupt is asserted and the corrupted data element is transferred to the Receiver Buffer Register.

Bit 5 = **PE**: *Parity Error Pending*.
This bit is set by hardware if the received word did not have the correct even or odd parity bit.
0: No Parity Error.
1: Parity Error occurred.

Bit 4 = **RXAP**: *Receiver Address Pending*.
RXAP is set by hardware after an interrupt acknowledged in the address mode.
0: No interrupt in address mode.
1: Interrupt in address mode occurred.

Note: The source of this interrupt is given by the couple of bits (AMEN, AM) as detailed in the IDPR register description.

Bit 3 = **RXBP**: *Receiver Break Pending bit*
This bit is set by hardware if the received data input is held low for the full word transmission time (start bit, data bits, parity bit, stop bit).
0: No break received.
1: Break event occurred.

Bit 2 = **RXDP**: *Receiver Data Pending bit.*
This bit is set by hardware when data is loaded into the Receiver Buffer Register.
0: No data received.
1: Data received in Receiver Buffer Register.

Bit 1 = **TXBEM**: *Transmitter Buffer register Empty*.
This bit is set by hardware if the Buffer Register is empty.
0: No Buffer Register empty event.
1: Buffer Register empty.

Bit 0 = **TXSEM**: *Transmitter Shift Register Empty*.
This bit is set by hardware if the Shift Register has completed the transmission of the available data.
0: No Shift Register empty event.
1: Shift register empty.

Note: The Interrupt Status Register bits can be reset but cannot be set by the user. The interrupt source must be cleared by resetting the related bit when executing the interrupt service routine (naturally the other pending bits should not be reset).

REGISTER DESCRIPTION (Cont'd)

RECEIVER BUFFER REGISTER (RXBR)

R248 - Read only

Reset value: undefined

7							0
RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0

Bit 7:0 = **RD[7:0]**: *Received Data.*
This register stores the data portion of the received word. The data will be transferred from the Receiver Shift Register into the Receiver Buffer Register at the end of the word. All receiver interrupt conditions will be updated at the time of transfer. If the selected character format is less than 8 bits, unused most significant bits will forced to "1".

Note: RXBR and TXBR are two physically different registers located at the same address.

TRANSMITTER BUFFER REGISTER (TXBR)

R248 - Write only

Reset value: undefined

7							0
TD7	TD6	TD5	TD4	TD3	TD2	TD1	TD0

Bit 7:0 = **TD[7:0]**: *Transmit Data*
The ST9 core will load the data for transmission into this register. The SCI will transfer the data from the buffer into the Shift Register when available. At the transfer, the Transmitter Buffer Register interrupt is updated. If the selected word format is less than 8 bits, the unused most significant bits are not significant.

Note: TXBR and RXBR are two physically different registers located at the same address.

INTERRUPT/DMA PRIORITY REGISTER (IDPR)

R249 - Read/Write

Reset value: undefined

7							0
AMEN	SB	SA	RXD	TXD	PRL2	PRL1	PRL0

Bit 7 = **AMEN**: *Address Mode Enable.*
This bit, together with the AM bit (in the CHCR register), decodes the desired addressing/9th data bit/character match operation.

In Address mode the SCI monitors the input serial data until its address is detected

AMEN	AM	
0	0	Address interrupt if 9th data bit = 1
0	1	Address interrupt if character match
1	0	Address interrupt if character match and 9th data bit =1
1	1	Address interrupt if character match with word immediately following Break

Note: Upon reception of address, the RXAP bit (in the Interrupt Status Register) is set and an interrupt cycle can begin. The address character will not be transferred into the Receiver Buffer Register but all data following the matched SCI address and preceding the next address word will be transferred to the Receiver Buffer Register and the proper interrupts updated. If the address does not match, all data following this unmatched address will not be transferred to the Receiver Buffer Register.

In any of the cases the RXAP bit must be reset by software before the next word is transferred into the Buffer Register.

When AMEN is reset and AM is set, a useful character search function is performed. This allows the SCI to generate an interrupt whenever a specific character is encountered (e.g. Carriage Return).

REGISTER DESCRIPTION (Cont'd)

Bit 6 = **SB**: *Set Break*.
0: Stop the break transmission after minimum break length.
1: Transmit a break following the transmission of all data in the Transmitter Shift Register and the Buffer Register.

Note: The break will be a low level on the transmitter data output for at least one complete word format. If software does not reset SB before the minimum break length has finished, the break condition will continue until software resets SB. The SCI terminates the break condition with a high level on the transmitter data output for one transmission clock period.

Bit 5 = **SA**: *Set Address*.
If an address/9th data bit mode is selected, SA value will be loaded for transmission into the Shift Register. This bit is cleared by hardware after its load.
0: Indicate it is not an address word.
1: Indicate an address word.

Note: Proper procedure would be, when the Transmitter Buffer Register is empty, to load the value of SA and then load the data into the Transmitter Buffer Register.

Bit 4 = **RXD**: *Receiver DMA Mask*.
This bit is reset by hardware when the transaction counter value decrements to zero. At that time a receiver End of Block interrupt can occur.
0: Disable Receiver DMA request (the RXDP bit in the ISR register can request an interrupt).
1: Enable Receiver DMA request (the RXDP bit in the ISR register can request a DMA transfer).

Bit 3 = **TXD**: *Transmitter DMA Mask*.
This bit is reset by hardware when the transaction counter value decrements to zero. At that time a transmitter End Of Block interrupt can occur.
0: Disable Transmitter DMA request (TXBEM or TXSEM bits in ISR can request an interrupt).
1: Enable Transmitter DMA request (TXBEM or TXSEM bits in ISR can request a DMA transfer).

Bit 2:0 = **PRL[2:0]**: *SCI Interrupt/DMA Priority bits*
The priority for the SCI is encoded with (PRL2,PRL1,PRL0). Priority level 0 is the highest, while level 7 represents no priority.

When the user has defined a priority level for the SCI, priorities within the SCI are hardware defined. These SCI internal priorities are:

Receiver DMA request	highest priority
Transmitter DMA request	
Receiver interrupt	
Transmitter interrupt	lowest priority

CHARACTER CONFIGURATION REGISTER (CHCR)

R250 - Read/Write

Reset value: undefined

7							0
AM	EP	PEN	AB	SB1	SB0	WL1	WL0

Bit 7 = **AM**: *Address Mode*.
This bit, together with the AMEN bit (in the IDPR register), decodes the desired addressing/9th data bit/character match operation. Please refer to the table in the IDPR register description.

Bit 6 = **EP**: *Even Parity*.
0: Select odd parity (when parity is enabled).
1: Select even parity (when parity is enabled).

Bit 5 = **PEN**: *Parity Enable*.
0: No parity bit.
1: Parity bit generated (transmit data) or checked (received data).

Note: If the address/9th bit is enabled, the parity bit will precede the address/9th bit (the 9th bit is never included in the parity calculation).

Bit 4 = **AB**: *Address/9th Bit*.
0: No Address/9th bit.
1: Address/9th bit included in the character format between the parity bit and the first stop bit. This bit can be used to address the SCI or as a ninth data bit.

REGISTER DESCRIPTION (Cont'd)

Bit 3:2 = **SB[1:0]**: *Number of Stop Bits*.

SB1	SB0	Number of stop bits	
		in 16X mode	in 1X mode
0	0	1	1
0	1	1.5	2
1	0	2	2
1	1	2.5	3

Bit 1:0 = **WL[1:0]**: *Number of Data Bits*

WL1	WL0	Data Length
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

CLOCK CONFIGURATION REGISTER (CCR)

R251 - Read/Write

Reset value: 0000 0000 (00h)

7							0
XTCLK	OCLK	XRX	XBRG	CD	AEN	LBEN	STPEN

Bit 7 = **XTCLK**
This bit, together with the OCLK bit, selects the source for the transmitter clock. The following table shows the coding of XTCLK and OCLK.

Bit 6 = **OCLK**
This bit, together with the XTCLK bit, selects the source for the transmitter clock. The following table shows the coding of XTCLK and OCLK.

XTCLK	OCLK	Pin Function
0	0	Pin is used as a general I/O
0	1	Pin = TXCLK (used as an input)
1	0	Pin = CLKOUT (outputs the Baud Rate Generator clock)
1	1	Pin = CLKOUT (outputs the Serial expansion and synchronous mode clock)

Bit 5 = **XRX**: *External Receiver Clock Source*
0: External receiver clock source not used.
1: Select the external receiver clock source.

Note: The external receiver clock frequency must be 16 times the data rate, or equal to the data rate, depending on the status of the CD bit.

Bit 4 = **XBRG**: *Baud Rate Generator Clock Source*.
0: Select INTCLK for the baud rate generator.
1: Select the external receiver clock for the baud rate generator.

Bit 3 = **CD**: *Clock Divisor*.
The status of CD will determine the SCI configuration (synchronous/asynchronous).
0: Select 16X clock mode for both receiver and transmitter.
1: Select 1X clock mode for both receiver and transmitter.

Note: In 1X clock mode, the transmitter will transmit data at one data bit per clock period. In 16X mode each data bit period will be 16 clock periods long.

Bit 2 = **AEN**: *Auto Echo Enable*.
0: No auto echo mode.
1: Put the SCI in auto echo mode.

Note: Auto Echo mode has the following effect: the SCI transmitter is disconnected from the data-out pin SOUT, which is driven directly by the receiver data-in pin, SIN. The receiver remains connected to SIN and is operational, unless loopback mode is also selected.

Bit 1 = **LBEN**: *Loopback Enable*.
0: No loopback mode.
1: Put the SCI in loopback mode.

Note: In this mode, the transmitter output is set to a high level, the receiver input is disconnected, and the output of the Transmitter Shift Register is looped back into the Receiver Shift Register input. All interrupt sources (transmitter and receiver) are operational.

REGISTER DESCRIPTION (Cont'd)

Bit 0 = **STPEN**: *Stick Parity Enable*.
0: The transmitter and the receiver will follow the parity of even parity bit EP in the CHCR register.
1: The transmitter and the receiver will use the opposite parity type selected by the even parity bit EP in the CHCR register.

EP	SPEN	Parity (Transmitter & Receiver)
0 (odd)	0	Odd
1 (even)	0	Even
0 (odd)	1	Even
1 (even)	1	Odd

BAUD RATE GENERATOR HIGH REGISTER (BRGHR)

R252 - Read/Write

Reset value: undefined

15							8
BG15	BG14	BG13	BG12	BG11	BG10	BG9	BG8

BAUD RATE GENERATOR LOW REGISTER (BRGLR)

R253 - Read/Write

Reset value: undefined

7							0
BG7	BG6	BG5	BG4	BG3	BG2	BG1	BG0

Bit 15:0 = *Baud Rate Generator MSB and LSB.*

The Baud Rate generator is a programmable divide by "N" counter which can be used to generate the clocks for the transmitter and/or receiver. This counter divides the clock input by the value in the Baud Rate Generator Register. The minimum baud rate divisor is 2 and the maximum divisor is 2^{16}-1. After initialization of the baud rate generator, the divisor value is immediately loaded into the counter. This prevents potentially long random counts on the initial load. If set to 0 or 1, the Baud Rate Generator is stopped.

SYNCHRONOUS INPUT CONTROL (SICR)

R254 - Read/Write

Reset value: 0000 0011 (03h)

7							0
SMEN	INPL	XCKPL	DCDEN	DCDPL	INPEN	X	X

Bit 7 = **SMEN**: *Synchronous Mode Enable*.
0: Disable all features relating to Synchronous mode (the contents of SICR and SOCR are ignored).
1: Select Synchronous mode with its programmed I/O configuration.

Bit 6 = **INPL**: *SIN Input Polarity*.
0: Polarity not inverted.
1: Polarity inverted.

Note: INPL only affects received data. In Auto-Echo mode SOUT = SIN even if INPL is set. In Loop-Back mode the state of the INPL bit is irrelevant.

Bit 5 = **XCKPL**: *Receiver Clock Polarity*.
0: RXCLK is active on the rising edge.
1: RXCLK is active on the falling edge.

Note: XCKPL only affects the receiver clock. In Auto-Echo mode CLKOUT = RXCLK independently of the XCKPL status. In Loop-Back the state of the XCKPL bit is irrelevant.

Bit 4 = **DCDEN**: *DCD Input Enable*.
0: Disable hardware synchronization.
1: Enable hardware synchronization.

Note: When DCDEN is set, RXCLK drives the receiver section only during the active level of the DCD input (DCD works as a gate on RXCLK, informing the MCU that a transmitting device is sending a synchronous frame to it).

Bit 3 = **DCDPL**: *DCD Input Polarity*.
0: The DCD input is active when LOW.
1: The DCD input is active when HIGH.

Note: DCDPL only affects the gating activity of the receiver clock. In Auto-Echo mode RTS = DCD independently of DCDPL. In Loop-Back mode, the state of DCDPL is irrelevant.

REGISTER DESCRIPTION (Cont'd)

Bit 2 = **INPEN**: *All Input Disable.*
0: Enable SIN/RXCLK/DCD inputs.
1: Disable SIN/RXCLK/DCD inputs.

Bit 1:0 = "Don't Care"

SYNCHRONOUS OUTPUT CONTROL (SOCR)

R255 - Read/Write

Reset value: 0000 0001 (01h)

7							0
OUTPL	OUTSB	OCKPL	OCKSB	RTSEN	RTSPL	-	X

Bit 7 = **OUTPL**: *SOUT Output Polarity.*
0: Polarity not inverted.
1: Polarity inverted.

Note: OUTPL only affects the data sent by the transmitter section. In Auto-Echo mode SOUT = SIN even if OUTPL=1. In Loop-Back mode, the state of OUTPL is irrelevant.

Bit 6 = **OUTSB**: *SOUT Output Stand-By Level*

0: SOUT stand-by level is HIGH.
1: SOUT stand-by level is LOW.

Bit 5 = **OCKPL**: *Transmitter Clock Polarity.*
0: CLKOUT is active on the rising edge.
1: CLKOUT is active on the falling edge.

Note: OCKPL only affects the transmitter clock. In Auto-Echo mode CLKOUT = RXCLK independently of the state of OCKPL. In Loop-Back mode the state of OCKPL is irrelevant.

Bit 4 = **OCKSB**: *Transmitter Clock Stand-By Level.*
0: The CLKOUT stand-by level is HIGH.
1: The CLKOUT stand-by level is LOW.

Bit 3 = **RTSEN**: *RTS Output Enable.*
0: Disable the RTS hardware synchronisation.
1: Enable the RTS hardware synchronization.

Note: When RTSEN is set, the RTS output becomes active just before the first active edge of CLKOUT and indicates to target device that the MCU is about to send a synchronous frame; it returns to its stand-by value just after the last active edge of CLKOUT (MSB transmitted).

Bit 2 = **RTSPL**: *RTS Output Polarity.*
0: The RTS output is active when LOW.
1: The RTS output is active when HIGH.

Note: RTSPL only affects the RTS activity on the output pin. In Auto-Echo mode RTS = DCD independently from the RTSPL value. In Loop-Back mode RTSPL value is 'Don't Care'.

Bit 1 = Reserved.

Bit 0 = "Don't Care"

9.6 EIGHT-CHANNEL ANALOG TO DIGITAL CONVERTER (ADC8)

9.6.1 Introduction

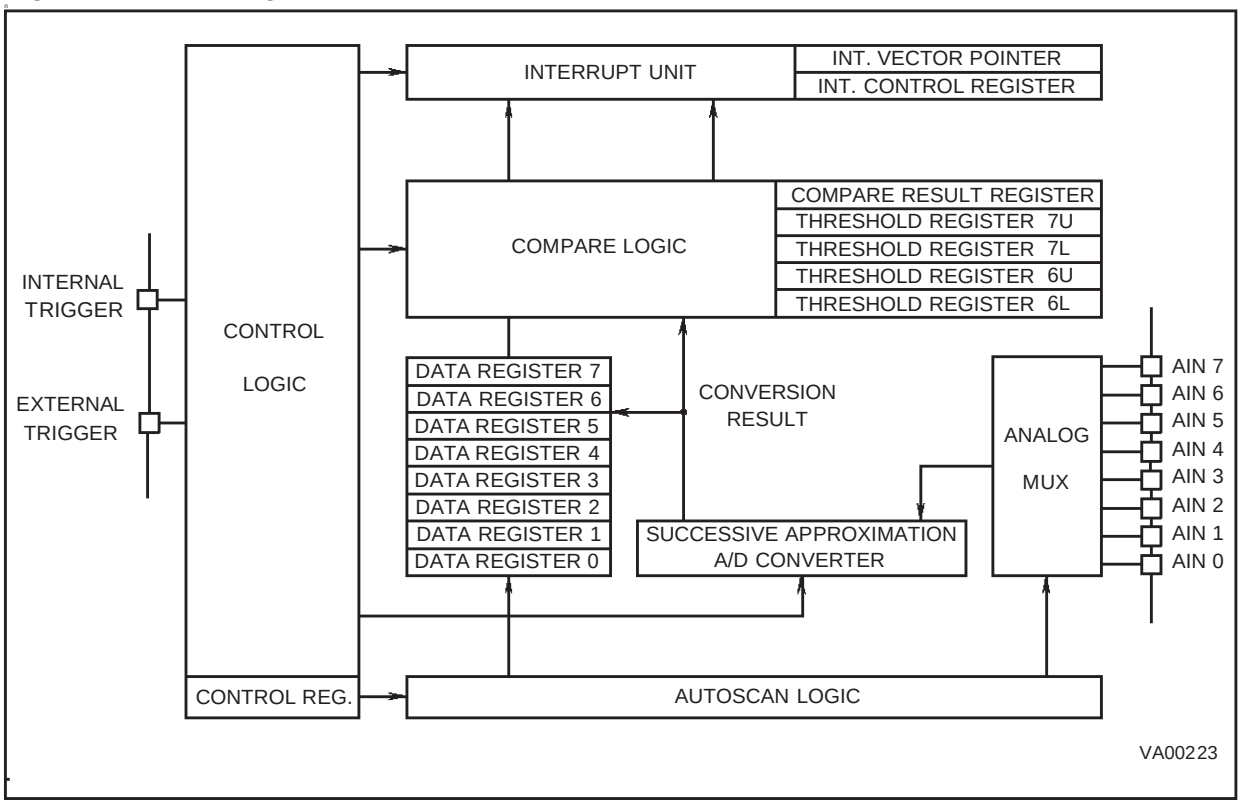
The 8-Channel Analog to Digital Converter (ADC8) comprises an input multiplex channel selector feeding a successive approximation converter. Conversion requires 138 INTCLK cycles (of which 87.5 are required for sampling), conversion time is thus a function of the INTCLK frequency; for instance, for a 20MHz clock rate, conversion of the selected channel requires 6.9µs. This time includes the 4.375µs required by the built-in Sample and Hold circuitry, which minimizes the need for external components and allows quick sampling of the signal to minimise warping and conversion error. Conversion resolution is 8 bits, with ±1 LSB maximum non-linearity error between V_{SS} and the analog V_{DD} reference.

The converter uses a fully differential analog input configuration for the best noise immunity and precision performance. Two separate supply references are provided to ensure the best possible supply noise rejection. In fact, the converted digital value, is referred to the analog reference voltage which determines the full scale converted value. Naturally**,** Analog and Digital V_{SS} MUST be common.

Up to 8 multiplexed Analog Inputs are available, depending on the specific device type. A group of signals can be converted sequentially by simply programming the starting address of the first analog channel to be converted and with the AUTOSCAN feature.

Two Analog Watchdogs are provided, allowing continuous hardware monitoring of two input channels. An Interrupt request is generated whenever the converted value of either of these two analog inputs is outside the upper or lower programmed threshold values. The comparison result is stored in a dedicated register.

Figure 83. Block Diagram



ANALOG TO DIGITAL CONVERTER(Cont'd)

Single and continuous conversion modes are available. Conversion may be triggered by an external signal or, internally, by the Multifunction Timer.

A Power-Down programmable bit allows the ADC8 to be set in low-power idle mode.

The ADC8's Interrupt Unit provides two maskable channels (Analog Watchdog and End of Conversion) with hardware fixed priority, and up to 7 programmable priority levels.

CAUTION: ADC8 INPUT PIN CONFIGURATION

The input Analog channel is selected by using the I/O pin Alternate Function setting (PXC2, PXC1, PXC0 = 1,1,1) as described in the I/O ports section. The I/O pin configuration of the port connected to the A/D converter is modified in order to prevent the analog voltage present on the I/O pin from causing high power dissipation across the input buffer. Deselected analog channels should also be maintained in Alternate function configuration for the same reason.

9.6.2 Functional Description

9.6.2.1 Operating Modes

Two operating modes are available: Continuous Mode and Single Mode. To enter one of these modes it is necessary to program the CONT bit of the Control Logic Register, the Continuous Mode is selected when CONT is set, while Single Mode is selected when CONT is reset.

Both modes operate in AUTOSCAN configuration, allowing sequential conversion of the input channels. The number of analog inputs to be converted may be set by software, by setting the number of the first channel to be converted into the Control Register (SC2, SC1, SC0 bits). As each conversion is completed, the channel number is automatically incremented, up to channel 7. For example, if SC2, SC1, SC0 are set to 0,1,1, conversion will proceed from channel 3 to channel 7, whereas, if SC2, SC1, SC0 are set to 1,1,1, only channel 7 will be converted.

When the ST bit of the Control Logic Register is set, either by software or by hardware (by an internal or external synchronisation trigger signal), the analog inputs are sequentially converted (from the first selected channel up to channel 7) and the results are stored in the relevant Data Registers.

In **Single Mode** (CONT = "0"), the ST bit is reset by hardware following conversion of channel 7; an End of Conversion (ECV) interrupt request is issued and the ADC8 waits for a new start event.

In **Continuous Mode** (CONT = "1"), a continuous conversion flow is initiated by the start event. When conversion of channel 7 is complete, conversion of channel 's' is initiated (where 's' is specified by the setting of the SC2, SC1 and SC0 bits); this will continue until the ST bit is reset by software. In all cases, an ECV interrupt is issued each time channel 7 conversion ends.

When channel 'i' is converted ('s' <'i' <7), the related Data Register is reloaded with the new conversion result and the previous value is lost. The End of Conversion (ECV) interrupt service routine can be used to save the current values before a new conversion sequence (so as to create signal sample tables in the Register File or in Memory).

9.6.2.2 Triggering and Synchronisation

In both modes, conversion may be triggered by internal or external conditions; externally this may be tied to EXTRG, as an Alternate Function input on an I/O port pin, and internally, it may be tied to INTRG, generated by a Multifunction Timer peripheral. Both external and internal events can be separately masked by programming the EXTG/ INTG bits of the Control Logic Register (CLR). The events are internally ORed, thus avoiding potential hardware conflicts. However, the correct procedure is to enable only one alternate synchronisation condition at any time.

The effect either of these synchronisation modes is to set the ST bit by hardware. This bit is reset, in Single Mode only, at the end of each group of conversions. In Continuous Mode, all trigger pulses after the first are ignored.

The synchronisation sources must be at a logic low level for at least the duration of one INTCLK cycle and, in Single Mode, the period between trigger pulses must be greater than the total time required for a group of conversions. If a trigger occurs when the ST bit is still set, i.e. when conversion is still in progress, it will be ignored.

On devices where two A/D Converters are present they can be triggered from the same source.

Converter	External Trigger	On Chip Event (Internal trigger)
A/D 0	EXTRG pin	MFT 0
A/D 1		

9.6.2.3 Analog Watchdogs

Two internal Analog Watchdogs are available for highly flexible automatic threshold monitoring of external analog signal levels.

ANALOG TO DIGITAL CONVERTER(Cont'd)

Analog channels 6 and 7 monitor an acceptable voltage level window for the converted analog inputs. The external voltages applied to inputs 6 and 7 are considered normal while they remain below their respective Upper thresholds, and above or at their respective Lower thresholds.

When the external signal voltage level is greater than, or equal to, the upper programmed voltage limit, or when it is less than the lower programmed voltage limit, a maskable interrupt request is generated and the Compare Results Register is updated in order to flag the threshold (Upper or Lower) and channel (6 or 7) responsible for the interrupt. The four threshold voltages are user programmable in dedicated registers (08h to 0Bh) of the ADC8 register page. Only the 4 MSBs of the Compare Results Register are used as flags (the 4 LSBs always return "1" if read), each of the four MSBs being associated with a threshold condition.

Following a hardware reset, these flags are reset. During normal ADC8 operation, the CRR bits are set, in order to flag an out of range condition and are automatically reset by hardware after a software reset of the Analog Watchdog Request flag in the ICR Register.

9.6.2.4 Power Down Mode

Before enabling an A/D conversion, the POW bit of the Control Logic Register must be set; this must be done at least 60µs before the first conversion start, in order to correctly bias the analog section of the converter circuitry.

When the ADC8 is not required, the POW bit may be reset in order to reduce the total power consumption. This is the reset configuration, and this state is also selected automatically when the ST9 is placed in Halt Mode (following the execution of the `halt` instruction).

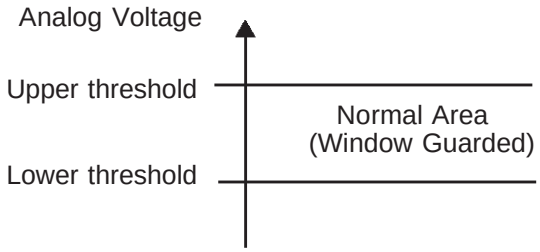
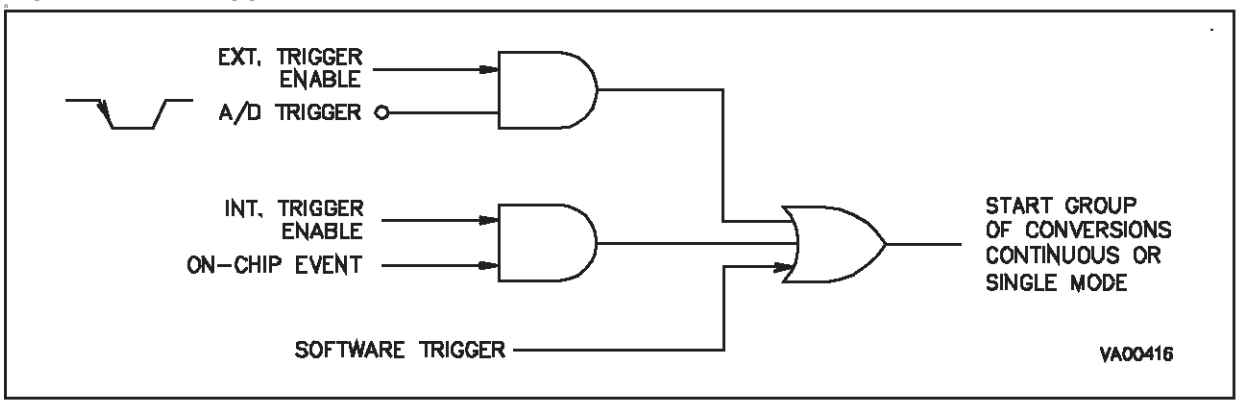
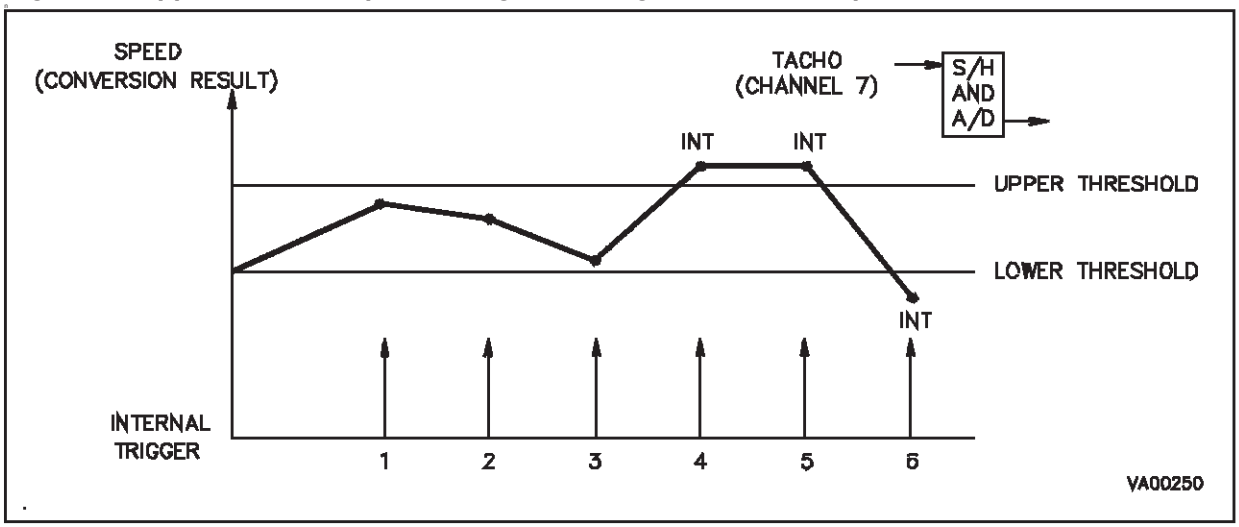


Figure 84. A/D Trigger Source



ANALOG TO DIGITAL CONVERTER(Cont'd)

Figure 85. Application Example: Analog Watchdog used in Motorspeed Control



9.6.3 Interrupts

The ADC8 provides two interrupt sources:

– End of Conversion

– Analog Watchdog Request

The A/D Interrupt Vector Register (IVR) provides hardware generated flags which indicate the interrupt source, thus allowing automatic selection of the correct interrupt service routine.

	7							0	
Analog Watchdog Request	X	X	X	X	X	X	0	0	Lower Word Address

	7							0	
End of Conv. Request	X	X	X	X	X	X	1	0	Upper Word Address

The A/D Interrupt vector should be programmed by the User to point to the first memory location in the Interrupt Vector table containing the base address of the four byte area of the interrupt vector table in which the address of the A/D interrupt service routines are stored.

The Analog Watchdog Interrupt Pending bit (AWD, ICR.6), is automatically set by hardware whenever any of the two guarded analog inputs go out of range. The Compare Result Register (CRR) tracks the analog inputs which exceed their programmed thresholds.

When two requests occur simultaneously, the Analog Watchdog Request has priority over the End of Conversion request, which is held pending.

The Analog Watchdog Request requires the user to poll the Compare Result Register (CRR) to determine which of the four thresholds has been exceeded. The threshold status bits are set to flag an out of range condition, and are automatically reset by hardware after a software reset of the Analog Watchdog Request flag in the ICR Register. The interrupt pending flags, ECV and AWD, should be reset by the user within the interrupt service routine. Setting either of these two bits by software will cause an interrupt request to be generated.

9.6.4 Register Description

DATA REGISTERS (DiR)

The conversion results for the 8 available channels are loaded into the 8 Data registers following conversion of the corresponding analog input.

CHANNEL 0 DATA REGISTER(D0R)
R240 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D0.7	D0.6	D0.5	D0.4	D0.3	D0.2	D0.1	D0.0

Bit 7:0 = D0.[7:0]: Channel 0 Data

CHANNEL 1 DATA REGISTER(D1R)
R241 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D1.7	D1.6	D1.5	D1.4	D1.3	D1.2	D1.1	D1.0

Bit 7:0 = D1.[7:0]: Channel 1 Data

CHANNEL 2 DATA REGISTER(D2R)
R242 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D2.7	D2.6	D2.5	D2.4	D2.3	D2.2	D2.1	D2.0

Bit 7:0 = D2.[7:0]: Channel 2 Data

CHANNEL 3 DATA REGISTER(D3R)
R243 - Read/Write
Register Page: 63
Reset Value: **undefined**

7							0
D3.7	D3.6	D3.5	D3.4	D3.3	D3.2	D3.1	D3.0

Bit 7:0 = D3.[7:0]: Channel 3 Data

CHANNEL 4 DATA REGISTER(D4R)
R244 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D4.7	D4.6	D4.5	D4.4	D4.3	D4.2	D4.1	D4.0

Bit 7:0 = D4.[7:0]: Channel 4 Data

CHANNEL 5 DATA REGISTER(D5R)
R245 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D5.7	D5.6	D5.5	D5.4	D5.3	D5.2	D5.1	D5.0

Bit 7:0 = D5.[7:0]: Channel 5 Data

CHANNEL 6 DATA REGISTER(D6R)
R246 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D6.7	D6.6	D6.5	D6.4	D6.3	D6.2	D6.1	D6.0

Bit 7:0 = D6.[7:0]: Channel 6 Data

CHANNEL 7 DATA REGISTER(D7R)
R247 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
D7.7	D7.6	D7.5	D7.4	D7.3	D7.2	D7.1	D7.0

Bit 7:0 = D7.[7:0]: Channel 7 Data

REGISTER DESCRIPTION (Cont'd)

LOWER THRESHOLD REGISTERS (LTiR)

The two Lower Threshold registers are used to store the user programmable lower threshold 8-bit values, to be compared with the current conversion results, thus setting the lower window limit.

CHANNEL 6 LOWER THRESHOLD REGISTER (LT6R)
R248 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
LT6.7	LT6.6	LT6.5	LT6.4	LT6.3	LT6.2	LT6.1	LT6.0

Bit 7:0 = LT6.[7:0]: Channel 6 Lower Threshold

CHANNEL 7 LOWER THRESHOLD REGISTER (LT7R)
R249 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
LT7.7	LT6.7	LT7.5	LT7.4	LT7.3	LT7.2	LT7.1	LT7.0

Bit 7:0 = LT7.[7:0]**:** *Channel 7 Lower Threshold*

UPPER THRESHOLD REGISTERS (UTIR)

The two Upper Threshold registers are used to store the user programmable upper threshold 8-bit values, to be compared with the current conversion results, thus setting the upper window limit.

CHANNEL 6 UPPER THRESHOLD REGISTER (UT6R)
R250 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
UT6.7	UT6.6	UT6.5	UT6.4	UT6.3	UT6.2	UT6.1	UT6.0

Bit 7:0 = **UT6.[7:0]**: *Channel 6 Upper Threshold value*

CHANNEL 7 UPPER THRESHOLD REGISTER (UT7R)
R251 - Read/Write
Register Page: 63
Reset Value: undefined

7							0
UT7.7	UT6.7	UT7.5	UT7.4	UT7.3	UT7.2	UT7.1	UT7.0

Bit 7:0 = **UT7.[7:0]**: *Channel 7 Upper Threshold value*

COMPARE RESULT REGISTER (CRR)
R252 - Read/Write
Register Page: 63
Reset Value: 0000 1111 (0Fh)

7							0
C7U	C6U	C7L	C6L	X	X	X	X

The result of the comparison between the current value of data registers 6 and 7 and the threshold registers is stored in this 4-bit register.

Bit 7 = **C7U**: *Compare Reg 7 Upper threshold*
Set when converted data is greater than or equal to the threshold value. Not affected otherwise.

Bit 6 = **C6U**: *Compare Reg 6 Upper threshold*
Set when converted data is greater than or equal to the threshold value. Not affected otherwise.

Bit 5 = **C7L**: *Compare Reg 7 Lower threshold*
Set when converted data is less than the threshold value. Not affected otherwise.

Bit 4 = **C6L**: *Compare Reg 6 Lower threshold*
Set when converted data is less than the threshold value. Not affected otherwise.

These bits should be reset at the end of the "Out of Range" interrupt service routine.

Bit 3:0 = undefined, returns "1" when read.

Note: Any software reset request of the ICR, will also cause all the compare status bits to forced by hardware to zero, in order to prevent possible overwriting if an interrupt request occurs between reset and the Interrupt request software reset.

REGISTER DESCRIPTION (Cont'd)

CONTROL LOGIC REGISTER (CLR)

The Control Logic Register (CLR) manages the ADC8's logic. Writing to this register will cause the current conversion to be aborted and the autoscan logic to be re-initialized. CLR is programmable as follows:

CONTROL LOGIC REGISTER (CLR)
R253 - Read/Write
Register Page: 63
Reset Value: 0000 0000 (00h)

7							0
SC2	SC1	SC0	EXTG	INTG	POW	CONT	ST

Bit 7:5 = **SC[2:0]:** *Start Conversion Address*.
These 3 bits define the starting analog input channel (Autoscan mode). The first channel addressed by SC[2:0] is converted, then the channel number is incremented for the successive conversion, until channel 7 (111) is converted. When SC2, SC1 and SC0 are all set, only channel 7 will be converted.

Bit 4 = **EXTG**: *External Trigger Enable*.
This bit is set and cleared by software.
0: External trigger disabled.
1: External trigger enabled. Allows a conversion sequence to be started on the subsequent edge of the external signal applied to the EXTRG pin (when enabled as an Alternate Function).

Bit 3 = **INTG**: *Internal Trigger Enable*.
This bit is set and cleared by software.
0: Internal trigger disabled.
1: Internal trigger enabled. Allows a conversion sequence to be started, synchronized by an internal signal (On-chip Event signal) from a Multifunction Timer peripheral.

Both External and Internal Trigger inputs are internally ORed, thus avoiding Hardware conflicts; however, the correct procedure is to enable only one alternate synchronization input at a time.

Bit 2 = **POW**: *Power Up/Power Down.*
This bit is set and cleared by software.
0: Power down mode: all power-consuming logic is disabled, thus selecting a low power idle mode.
1: Power up mode: the A/D converter logic and analog circuitry is enabled.

Bit 1 = **CONT**: *Continuous/Single*.
0: Single Mode: a single sequence of conversions is initiated whenever an external (or internal) trigger occurs, or when the ST bit is set by software.
1: Continuous Mode: the first sequence of conversions is started, either by software (by setting the ST bit), or by hardware (on an internal or external trigger, depending on the setting of the INTG and EXTG bits); a continuous conversion sequence is then initiated.

Note: The effect of the either synchronization mode is to set the START/STOP bit, which is reset by hardware when in SINGLE mode, at the end of each sequence of conversions.

Requirements: The External Synchronisation Input must receive a low level pulse wider than an INTCLK period and, for both External and On-Chip Event synchronisation, the repetition period must be greater than the time required for the selected sequence of conversions.

Bit 0 = **ST**: *Start/Stop.*
0: Stop conversion. When the A/D converter is running in Single Mode, this bit is hardware reset at the end of a sequence of conversions.
1: Start a sequence of conversions.

REGISTER DESCRIPTION (Cont'd)

INTERRUPT CONTROL REGISTER (ICR)

The Interrupt Control Register contains the three priority level bits, the two source flags, and their bit mask:

INTERRUPT CONTROL REGISTER(ICR)
R254 - Read/Write
Register Page: 63
Reset Value: 0000 1111 (0Fh)

7							0
ECV	AWD	ECI	AWDI	X	PL2	PL1	PL0

Bit 7 = **ECV**: *End of Conversion.*
This bit is automatically set by hardware after a group of conversions is completed. It must be reset by the user, before returning from the Interrupt Service Routine. Setting this bit by software will cause a software interrupt request to be generated.
0: No End of Conversion event occurred
1: An End of Conversion event occurred

Bit 6 = **AWD**: *Analog Watchdog.*
This is automatically set by hardware whenever either of the two monitored analog inputs goes out of bounds. The threshold values are stored in registers F8h and FAh for channel 6, and in registers F9h and FBh for channel 7 respectively. The Compare Result Register (CRR) keeps track of the analog inputs exceeding the thresholds.

The AWD bit must be reset by the user, before returning from the Interrupt Service Routine. Setting this bit by software will cause a software interrupt request to be generated.
0: No Analog Watchdog event occurred
1: An Analog Watchdog event occurred

Bit 5 = **ECI**: *End of Conversion Interrupt Enable.*

This bit masks the End of Conversion interrupt request.
0: Mask End of Conversion interrupts
1: Enable End of Conversion interrupts

Bit 4 = **AWDI**: *Analog Watchdog Interrupt Enable.*
This bit masks or enables the Analog Watchdog interrupt request.
0: Mask Analog Watchdog interrupts
1: Enable Analog Watchdog interrupts

Bit 3 = Reserved.

Bit 2:0 = **PL[2:0]:** *A/D Interrupt Priority Level*

These three bits allow selection of the Interrupt priority level for the ADC8.

INTERRUPT VECTOR REGISTER (IVR)
R255 - Read/Write
Register Page: 63
Reset Value: xxxx xx10 (x2h)

7							0
V7	V6	V5	V4	V3	V2	W1	0

Bit 7:2 = **V[7:2]:** *A/D Interrupt Vector.*

This vector should be programmed by the User to point to the first memory location in the Interrupt Vector table containing the starting addresses of the A/D interrupt service routines.

Bit 1 = **W1**: *Word Select.*

This bit is set and cleared by hardware, according to the A/D interrupt source.
0: Interrupt source is the Analog Watchdog, pointing to the lower word of the A/D interrupt service block (defined by V[7:2]).
1:Interrupt source is the End of Conversion interrupt, thus pointing to the upper word.

Note: When two requests occur simultaneously, the Analog Watchdog Request has priority over the End of Conversion request, which is held pending.

Bit 0 = Reserved. Forced by hardware to 0.

10 ELECTRICAL CHARACTERISTICS

This product contains devices to protect the inputs against damage due to high static voltages, however it is advisable to take normal precaution to avoid application of any voltage higher than the specified maximum rated voltages.

For proper operation it is recommended that V_I and V_O be higher than V_{SS} and lower than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriate logic voltage level (V_{DD} or V_{SS}).

Power Considerations: The average chip-junction temperature, T_J, in Celsius can be obtained from:

T_J= TA + PD x RthJA

Where: T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

P_D = P_{INT} + P_{PORT}.

P_{INT} = I_{DD} x V_{DD} (chip internal power).

P_{PORT} =Port power dissipation determined by the user)

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	– 0.3 to 7.0	V
AV_{DD}	A/D Converter Analog Reference	V_{DD} -0.3 to V_{DD} + 0.3	V
AV_{SS}	A/D Converter V_{SS}	V_{SS}	
V_I	Input Voltage	– 0.3 to V_{DD} +0.3	V
V_{AIN}	Analog Input Voltage (A/D Converter)	V_{SS} -0.3 to V_{DD} + 0.3 V_{SSA} -0.3 to V_{DDA} + 0.3	V
V_O	Output Voltage	– 0.3 to V_{DD} +0.3	V
T_{STG}	Storage Temperature	– 55 to + 150	°C
I_{INJ}	Pin Injection Current Digital and Analog Input	-5 to +5	mA
	Maximum Accumulated Pin injection Current in the device	-50 to +50	mA

Note: Stresses above those listed as “absolute maximum ratings“ may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability. All voltages are referenced to V_{SS}

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Value		Unit
		Min.	Max.	
T_A	Operating Temperature	-40	85	°C
V_{DD}	Operating Supply Voltage	4.5	5.5	V
f_{INTCLK}	Internal Clock Frequency @ 4.5V - 5.5V Internal Clock Frequency @ 2.7V - 3.3V	0(1)	16 12	MHz

Note 1. 1MHz when A/D is used

PACKAGE THERMAL CHARACTERISTICS

Symbol	Parameter	Package	Value			Unit
			Min.	Typ.	Max.	
Rth	Thermal junction to ambiant	PLCC84		35		°C/W
		PQFP80		40		

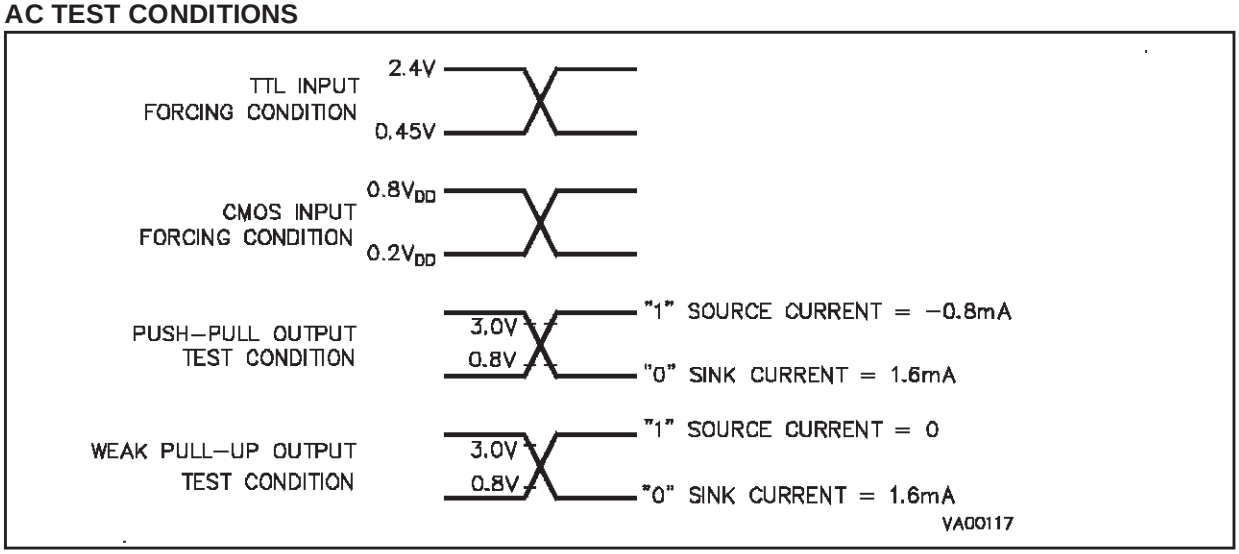
DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10% T_A = -40°C + 85°C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IHCK}	Clock Input High Level	External Clock	0.7 V_{DD}		V_{DD} + 0.3	V
V_{ILCK}	Clock Input Low Level	External Clock	– 0.3		0.3 V_{DD}	V
V_{IH}	Input High Level	TTL	2.0		V_{DD} + 0.3	V
		CMOS	0.7 V_{DD}		V_{DD} + 0.3	V
V_{IL}	Input Low Level	TTL	– 0.3		0.8	V
		CMOS	– 0.3		0.3 V_{DD}	V
V_{IHRS}	$\overline{\text{RESET}}$ Input High Level		0.7 V_{DD}		V_{DD} + 0.3	V
V_{ILRS}	$\overline{\text{RESET}}$ Input Low Level		–0.3		0.3 V_{DD}	V
V_{HYRS}	$\overline{\text{RESET}}$ Input Hysteresis		0.3		1.5	V
V_{OH}	Output High Level	Push Pull, Iload = – 0.8mA	V_{DD} – 0.8			V
V_{OL}	Output Low Level	Push Pull or Open Drain, Iload = 1.6mA			0.4	V
I_{WPU}	Weak Pull-up Current	Bidirectional Weak Pull-up, V_{OL} = 0V	– 50	– 200	– 420	µA
I_{LKIO}	I/O Pin Input Leakage	Input/Tri-State, 0V < V_{IN} < V_{DD}	– 10		+ 10	µA
I_{LKRS}	$\overline{\text{RESET}}$ Pin Input Leakage	0V < V_{IN} < V_{DD}	– 30		+ 30	µA
$I_{LKA/D}$	A/D Conv. Input Leakage		– 3		+ 3	µA
I_{LKAP}	Active Pull-up Input Leakage	0V < V_{IN} < 0.8V	– 10		+ 10	µA
I_{LKOS}	OSCIN Pin Input Leakage	0V < V_{IN} < V_{DD}			±3	µA

Note: All I/O Ports are configured in bidirectional weak pull-up mode with no DC load external clock pin (OSCIN) is driven by square wave external clock. No peripheral working.

AC TEST CONDITIONS



AC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5V ± 10% T_A = -40°C + 85°C, unless otherwise specified)

Symbol	Parameter	INTCLK	VDD				Unit
			Typ.		Max.		
			ROM	OTP	ROM	OTP	
I_{DDRUN}	Run Mode Current	25MHz	45	67	52.5	75	mA
$I_{DDPLLRUN}$	Run Mode Current		1.8	2.7	2.1	3	mA per MHz
I_{DDWFI}	WFI Mode Current	25MHz	15	20			mA
$I_{DDLPWFI}$	Low Power WFI Mode Current	5MHz/64	1.0	1.3			mA
I_{HALT}	HALT Mode Current		10		10		µA

Note: All I/O Ports are configured in bidirectional weak pull-up mode with no DC load, external clock pin (OSCIN) is driven by square wave external clock

EXTERNAL BUS TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = -40°C + 85°C, Cload = 50pF,INTCLK = 16MHz, unless otherwise specified)

N°	Symbol	Parameter	Value (Note)			Unit
			Formula	Min.	Max.	
1	TsA (AS)	Address Set-up Time before $\overline{AS}$ ↑	Tck*Wa+TckH-9	23		ns
2	ThAS (A)	Address Hold Time after $\overline{AS}$ ↑	TckL-4	28		ns
3	TdAS (DR)	$\overline{AS}$ ↑ to Data Available (read)	Tck*(Wd+1)+3		65	ns
4	TwAS	$\overline{AS}$ Low Pulse Width	Tck*Wa+TckH-5	27		ns
5	TdAz (DS)	Address Float to $\overline{DS}$ ↓	0	0		ns
6	TwDS	$\overline{DS}$ Low Pulse Width	Tck*Wd+TckH-5	27		ns
7	TdDSR (DR)	$\overline{DS}$ ↓ to Data Valid Delay (read)	Tck*Wd+TckH+4		35	ns
8	ThDR (DS)	Data to $\overline{DS}$ ↑ Hold Time (read)	7	7		ns
9	TdDS (A)	$\overline{DS}$ ↑ to Address Active Delay	TckL+11	43		ns
10	TdDS (AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	TckL-4	28		ns
11	TsR/W (AS)	R/$\overline{W}$ Set-up Time before $\overline{AS}$ ↑	Tck*Wa+TckH-17	15		ns
12	TdDSR (R/W)	$\overline{DS}$ ↑ to R/$\overline{W}$ and Address Not Valid Delay	TckL-1	31		ns
13	TdDW (DSW)	Write Data Valid to $\overline{DS}$ ↓ Delay	-16	-16		ns
14	TsD(DSW)	Write Data Set-up before $\overline{DS}$ ↑	Tck*Wd+TckH-16	16		ns
15	ThDS (DW)	Data Hold Time after $\overline{DS}$ ↑ (write)	TckL-3	29		ns
16	TdA (DR)	Address Valid to Data Valid Delay (read)	Tck*(Wa+Wd+1)+TckH-7		86	ns
17	TdAs (DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay	TckL-6	26		ns

Note: The value in the left hand column shows the formula used to calculate the timing minimum or maximum from the oscillator clock period, prescale value and number of wait cycles inserted.
The value in the right hand two columns show the timing minimum and maximum for an external clock at 24 MHz divided by 2, prescaler value of zero and zero wait status.

Legend:
Tck = INTCLK period = OSCIN period when OSCIN is not divided by 2;
2*OSCIN period when OSCIN is divided by 2;
OSCIN period / PLL factor when the PLL is enabled

TckH = INTCLK high pulse width (normally = Tck/2, except when INTCLK = OSCIN, in which case it is OSCIN high pulse width)

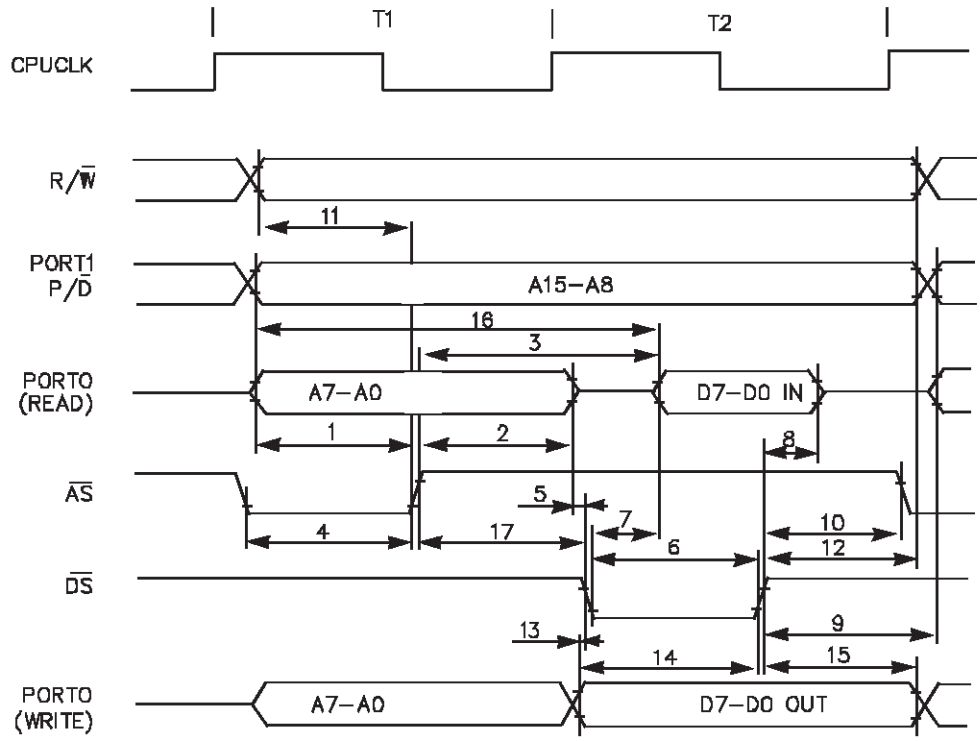
TckL = INTCLK low pulse width (normally = Tck/2, except when INTCLK = OSCIN, in which case it is OSCIN low pulse width)

P = clock prescaling value (=PRS; division factor = 1+P)

Wa = wait cycles on $\overline{AS}$; = max (P, programmed wait cycles in EMR2, requested wait cycles with $\overline{WAIT}$)

Wd = wait cycles on $\overline{DS}$; = max (P, programmed wait cycles in WCR, requested wait cycles with $\overline{WAIT}$)

EXTERNAL BUS TIMING



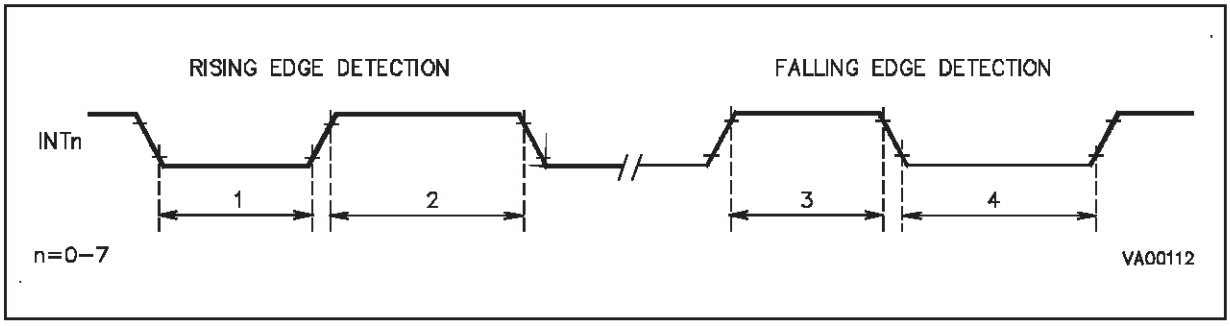
EXTERNAL INTERRUPT TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = -40°C +85°C, Cload = 50pF, INTCLK = 12MHz, Push-pull output configuration, unless otherwise specified)

N°	Symbol	Parameter	**Value** (Note)				Unit
			OSCIN Divided by 2 Min.	**OSCIN Not Divided by 2 Min.**	**Min.**	**Max.**	
1	TwLR	Low Level Minimum Pulse Width in Rising Edge Mode	2TpC+12	TpC+12	95		ns
2	TwHR	High Level Minimum Pulse Width in Rising Edge Mode	2TpC+12	TpC+12	95		ns
3	TwHF	High Level Minimum Pulse Width in Falling Edge Mode	2TpC+12	TpC+12	95		ns
4	TwLF	Low Level Minimum Pulse Width in Falling Edge Mode	2TpC+12	TpC+12	95		ns

Note: The value left hand two columns show the formula used to calculate the timing minimum or maximum from the oscillator clock period, prescale value and number of wait cycles inserted.
The value right hand two columns show the timing minimum and maximum for an external clock at 24 MHz divided by 2, prescale value of zero and zero wait status.

EXTERNAL INTERRUPT TIMING



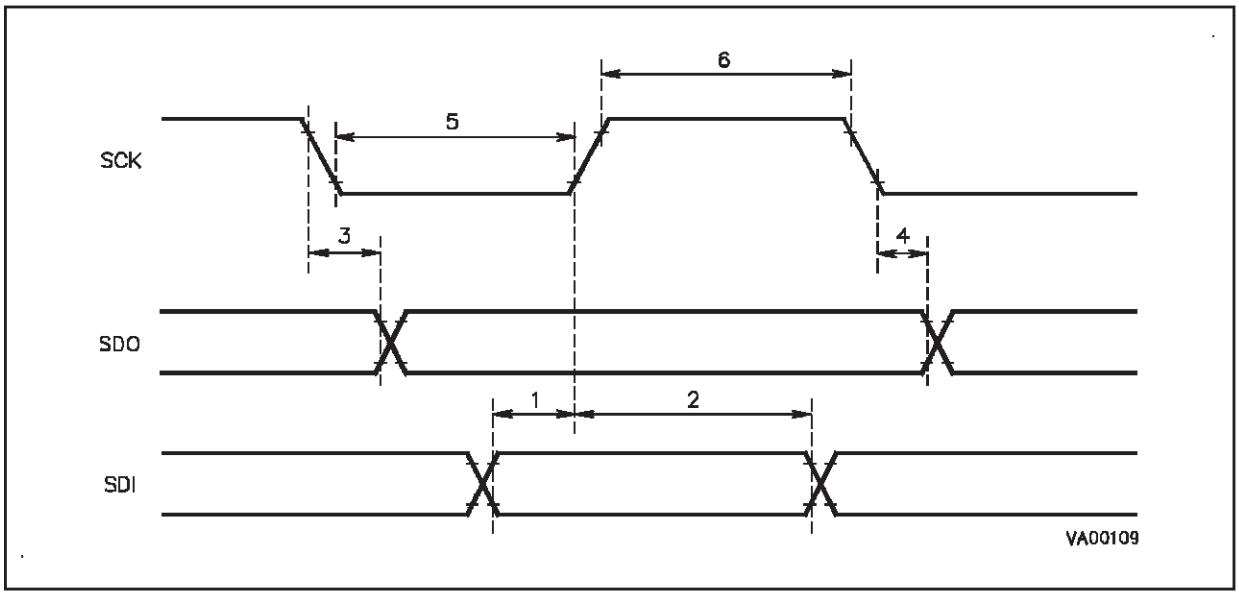
SPI TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = -40°C + 85°C, Cload = 50pF, INTCLK = 12MHz, Output Alternate Function set as Push-pull)

N°	Symbol	Parameter	Value		Unit
			Min.	Max.	
1	TsDI	Input Data Set-up Time	100		ns
2	ThDI (1)	Input Data Hold Time	1/2 TpC+100		ns
3	TdOV	SCK to Output Data Valid		100	ns
4	ThDO	Output Data Hold Time	-20		ns
5	TwSKL	SCK Low Pulse Width	300		ns
6	TwSKH	SCK High Pulse Width	300		ns

Note: TpC is the OSCIN Clock period.

SPI TIMING

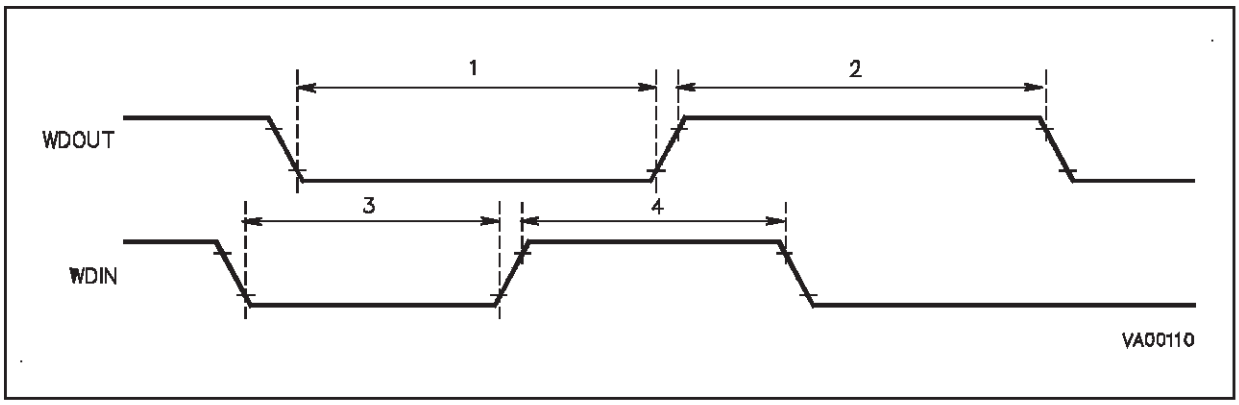


WATCHDOG TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = -40°C + 85°C, Cload = 50pF, INTCLK = 12MHz, Push-pull output configuration, unless otherwise specified)

N°	Symbol	Parameter	Values		Unit
			Min.	Max.	
1	TwWDOL	WDOUT Low Pulse Width	620		ns
2	TwWDOH	WDOUT High Pulse Width	620		ns
3	TwWDIL	WDIN High Pulse Width	350		ns
4	TwWDIH	WDIN Low Pulse Width	350		ns

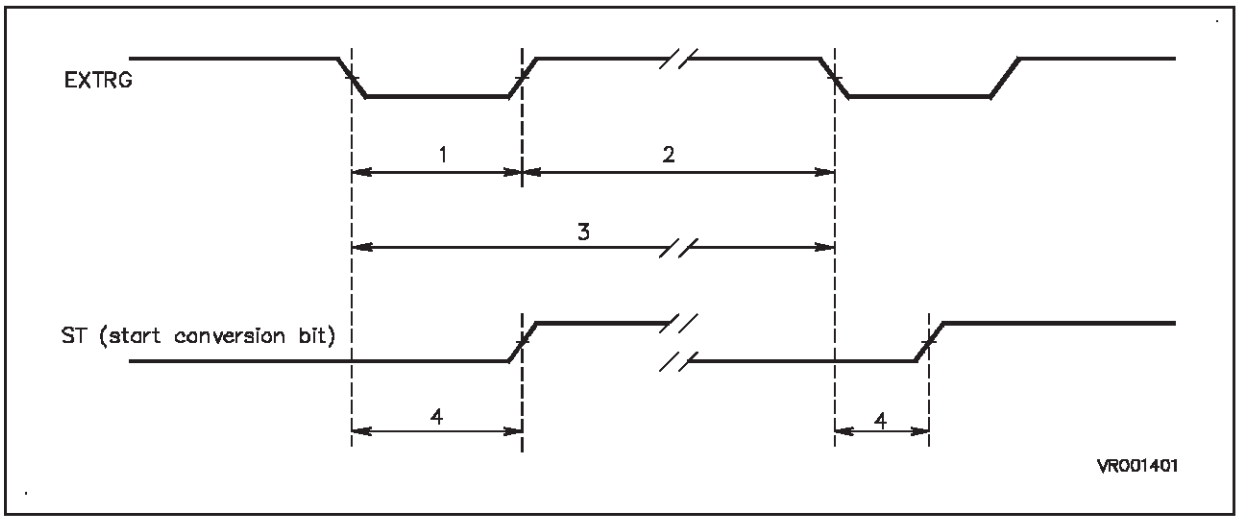
WATCHDOG TIMING



A/D EXTERNAL TRIGGER TIMING TABLE

N°	Symbol	Parameter	OSCIN Divided by 2 (2)		OSCIN Not Divided by 2 (2)		Value (3)		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
1	Tw_{LOW}	External trigger pulse width	2 x Tpc		Tpc		83	-	ns
2	Tw_{HIGH}	External trigger pulse distance	2 x Tpc		Tpc		83	-	ns
3	Tw_{EXT}	External trigger active edges distance (1)	276n x Tpc		138n x Tpc		n x 11.5	-	µs
4	Td_{STR}	EXTRG falling edge and first conversion start	Tpc	3 x Tpc	.5 x Tpc	1.5 x Tpc	41.5	125	ns

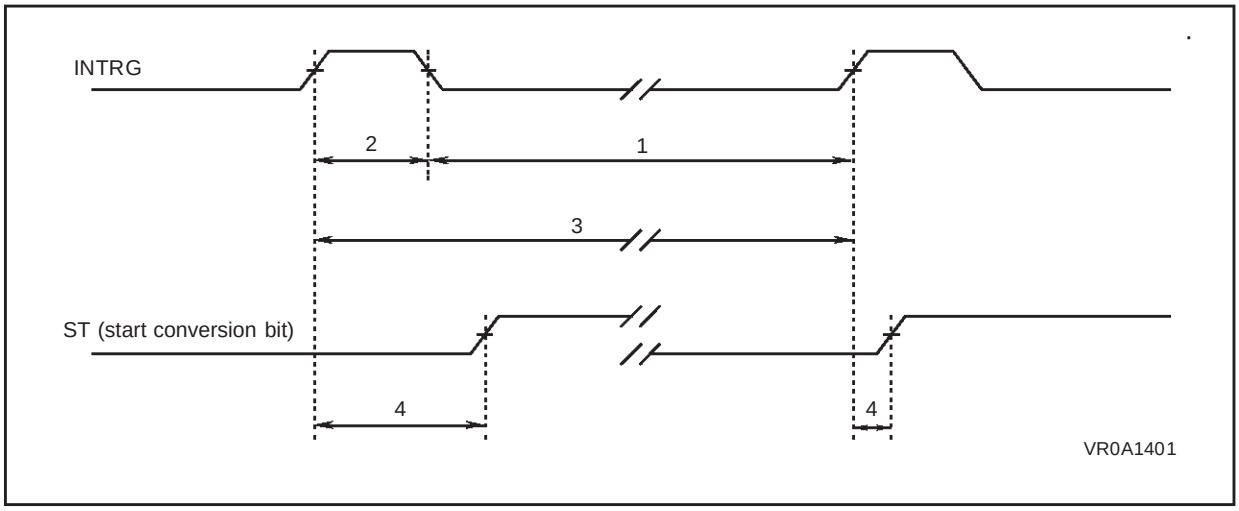
A/D EXTERNAL TRIGGER TIMING



A/D INTERNAL TRIGGER TIMING TABLE

N°	Symbol	Parameter	OSCIN Divided by 2 (2)		OSCIN Not Divided by 2 (2)		Value (3)		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
1	Tw_{HIGH}	Internal trigger pulse width	Tpc		.5 x Tpc		41.5	-	ns
2	Tw_{LOW}	Internal trigger pulse distance	6 x Tpc		3 x Tpc		250	-	ns
3	Tw_{EXT}	Internal trigger active edges distance (1)	276n x Tpc		138n x Tpc		n x 11.5	-	µs
4	Tw_{STR}	Internal delay between INTRG rising edge and first conversion start	Tpc	3 x Tpc	.5 x Tpc	1.5 x Tpc	41.5	125	ns

A/D INTERNAL TRIGGER TIMING

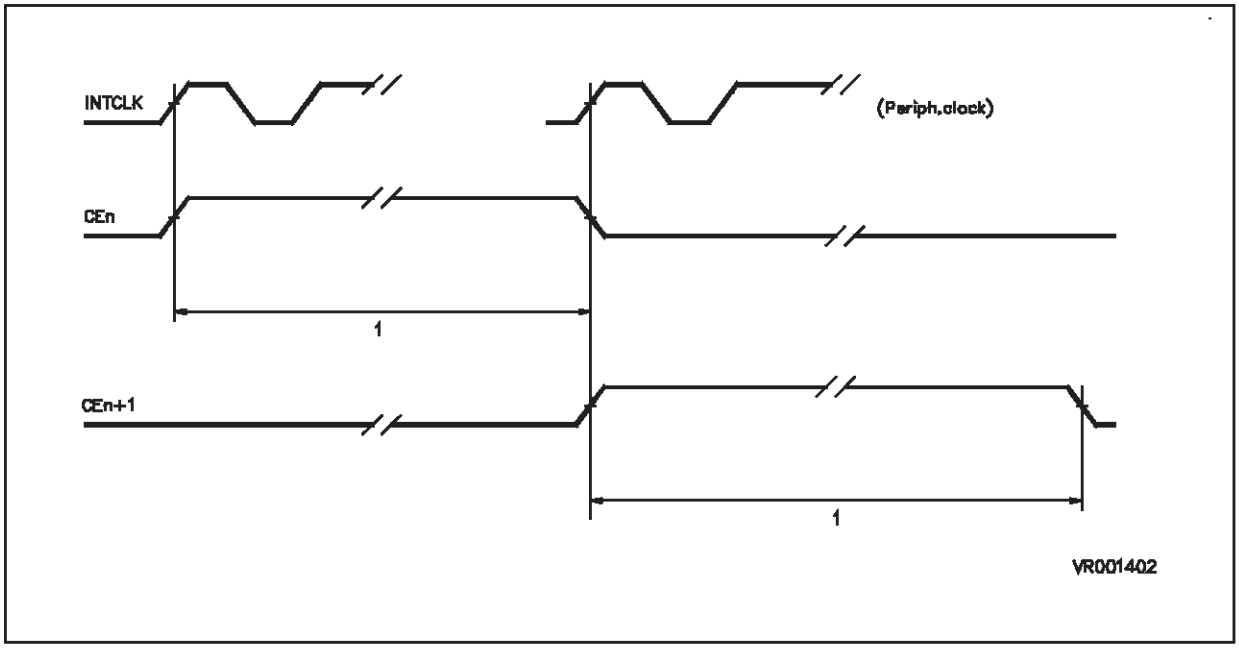


A/D CHANNEL ENABLE TIMING TABLE

N°	Symbol	Parameter	OSCIN Divided by 2 (2)		OSCIN Not Divided by 2 (2)		Value (3)		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
1	Tw_{EXT}	CEn Pulse width (1)	276n x Tpc		138n x Tpc		n x 11.5	-	µs

Notes:
1. n = number of autoscanned channels (1 < n < 8)
2. Variable clock (Tpc = OSCIN clock period)
3. INTCLK = 12MHz

A/D CHANNEL ENABLE TIMING



A/D ANALOG SPECIFICATIONS (VDD = 4.5V TO 5.0 V)

Parameter	Typical	Minimum	Maximum	Units (1)	Notes
Conversion time		138		INTCLK	(2)
Sample time		87.51		INTCLK	
Power-up time		60		µs	
Resolution	8	8		µs	
Monotonicity	GUARANTEED	+		bits	
No missing codes	GUARANTEED	+			
Zero input reading		00		Hex	
Full scale reading			FF	Hex	
Offset error	.5		1	LSBs	(1,4)
Gain error	.5		1	LSBs	(4)
Diff. Non Linearity	±.3	±.2	±.5	LSBs	(4)
Int. Non Linearity			1	LSBs	(4)
Absolute Accuracy			1	LSBs	(4)
A_{VCC}/A_{VSS} Resistance	13.5	16	11	KΩ	
Input Resistance	12	8	15	KΩ	(3)
Hold Capacitance			30	pF	
Input Leakage			±3	µA	

Notes:

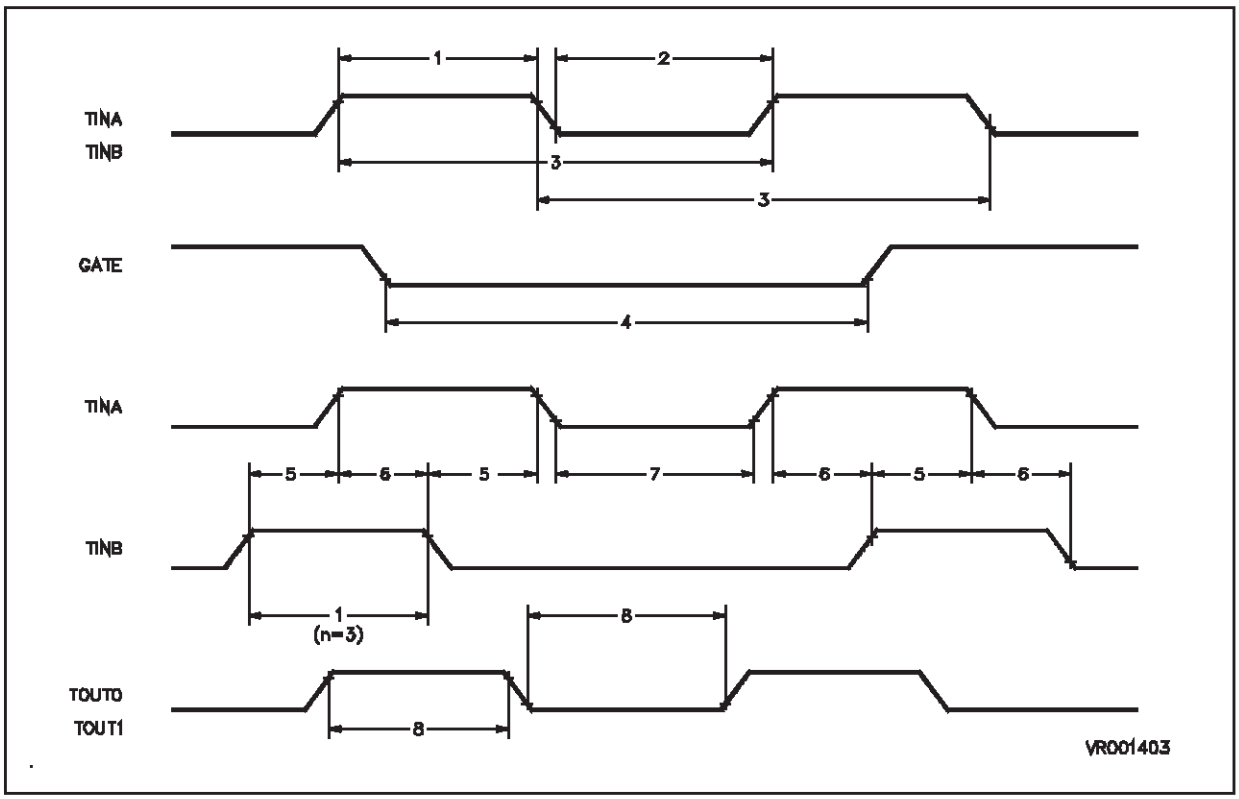
1. "LSBs", as used here, has a value of AVDD/256
2. Including sample time
3. It must be intended as the internal series resistance before the sampling capacitor
4. This is a typical expected value, but not a tested production parameter.
   If V(i) is the value of the i-th transition level (0 < i < 254), the performance of the A/D converter has been valued as follows:
   OFFSET ERROR= deviation between the actual V(0) and the ideal V(0) (=1/2 LSB)
   GAIN ERROR= deviation between the actual V(254) and the ideal V(254) (=AVCC-3/2 LSB)
   DNL ERROR= max {[V(i) - V(i-1)]/LSB - 1}
   INL ERROR= max {[V(i) - V(0)]/LSB - i}
   ABS. ACCURACY= overall max conversion error
   S/N ratio has been valued by sampling a sinusoidal input waveform and then calculating its Fast Fourier Transform.

MULTIFUNCTION TIMER UNIT EXTERNAL TIMING TABLE

N°	Symbol	Parameter	OSCIN Divided by 2 (3)	OSCIN Not Divided by 2 (3)	Value (4)		Unit	Note
					Min.	Max.		
1	Tw_{CTW}	External clock/trigger pulse width	2n x Tpc	n x Tpc	n x 83	-	ns	1
2	Tw_{CTD}	External clock/trigger pulse distance	2n x Tpc	n x Tpc	n x 83	-	ns	1
3	Tw_{AED}	Distance between two active edges	6 x Tpc	3 x Tpc	249	-	ns	
4	Tw_{GW}	Gate pulse width	12 x Tpc	6 x Tpc	498	-	ns	
5	Tw_{LBA}	Distance between TINB pulse edge and the following TINA pulse edge	2 x Tpc	Tpc	83	-	ns	2
6	Tw_{LAB}	Distance between TINA pulse edge and the following TINB pulse edge	0		0	-	ns	2
7	Tw_{AD}	Distance between two TxINA pulses	0		0	-	ns	2
8	Tw_{OWD}	Minimum output pulse width/distance	6 x Tpc	3 x Tpc	249	-	ns	

Notes:
1. n = 1 if the input is rising OR falling edge sensitive
   n = 3 if the input is rising AND falling edge sensitive
2. In Autodiscrimination mode
3. Variable clock (Tpc = OSCIN period)
4. INTCLK = 12 MHz

MULTIFUNCTION TIMER UNIT EXTERNAL TIMING



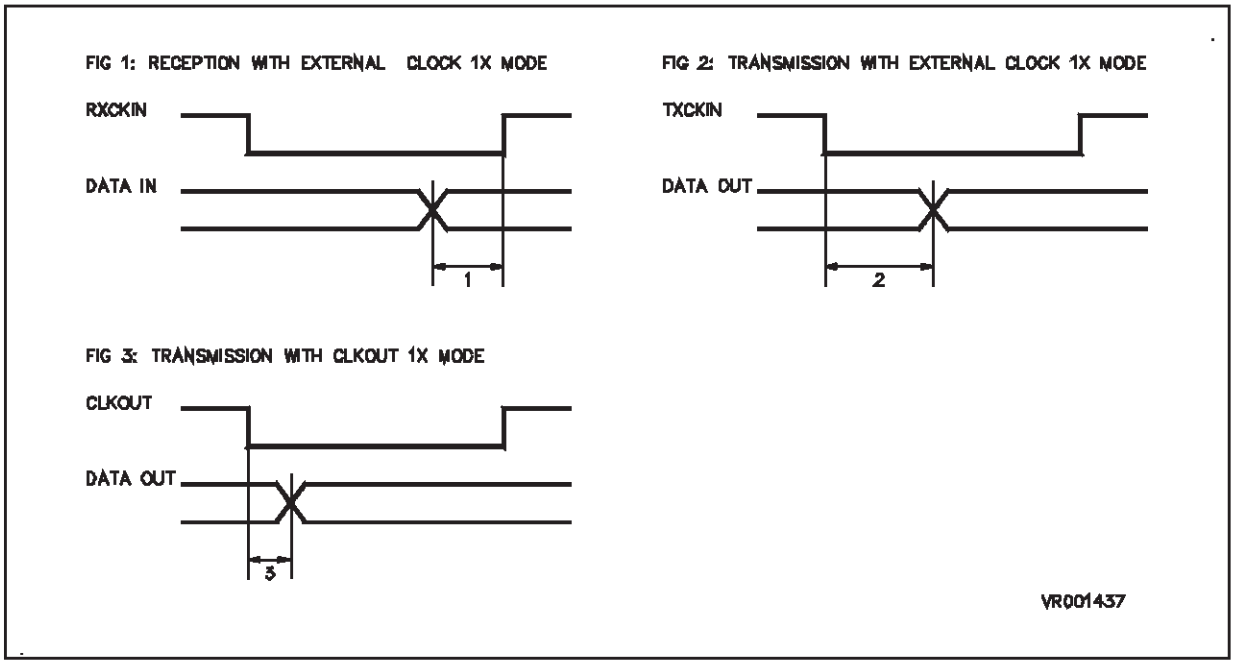
SCI TIMING TABLE

(V_{DD} = 5V ± 10%, T_A = - 40°C to +85°C, Cload = 50pF, INTCLK = 12MHz, Output Alternate Function set as Push-pull)

N°	Symbol	Parameter	Condition	Value		Unit
				Min.	Max.	
	F_{RxCKIN}	Frequency of RxCKIN	1 x mode		$F_{CK}/8$	Hz
			16 x mode		$F_{CK}/4$	Hz
	Tw_{RxCKIN}	RxCKIN shortest pulse	1 x mode	4 T_{CK}		s
			16 x mode	2 T_{CK}		s
	F_{TxCKIN}	Frequency of TxCKIN	1 x mode		$F_{CK}/8$	Hz
			16 x mode		$F_{CK}/4$	Hz
	Tw_{TxCKIN}	TxCKIN shortest pulse	1 x mode	4 T_{CK}		s
			16 x mode	2 T_{CK}		s
1	Ts_{DS}	DS (Data Stable) before rising edge of RxCKIN	1 x mode reception with RxCKIN	$T_{PC}/2$		ns
2	Td_{D1}	TxCKIN to Data out delay Time	1 x mode transmission with external clock C load <100pF		2.5 T_{PC}	ns
3	Td_{D2}	CLKOUT to Data out delay Time	1 x mode transmission with CLKOUT	350		ns

Note: FCK = 1/TCK

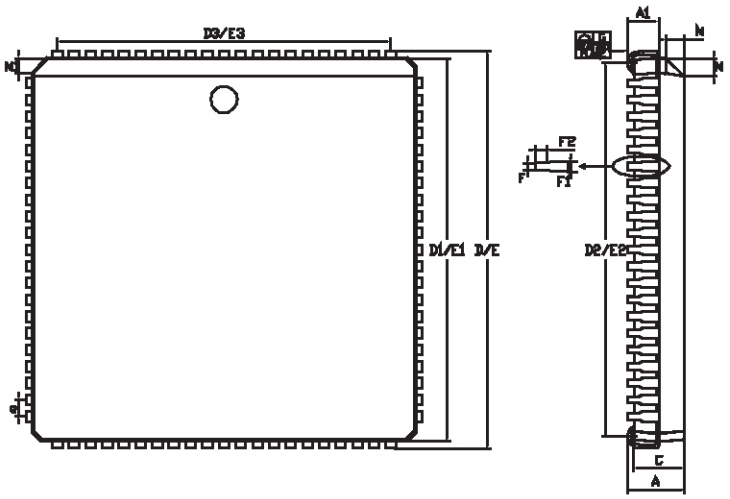
SCI TIMING



11 GENERAL INFORMATION

11.1 PACKAGE MECHANICAL DATA

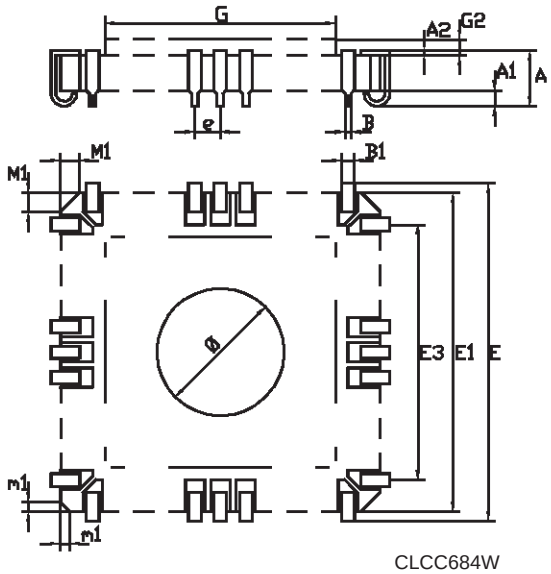
84-PIN PLASTIC LEADED CHIP CARRIER PACKAGE



PLCC84

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	4.19		0.51	0.165		0.020
A1	2.29		3.30	0.090		0.130
D/E	30.10		30.35	1.185		1.195
D1/E1	29.21		29.41	1.150		1.158
D2/E2	27.69		28.70	1.090		1.130
D3/E3		25.40			1.000	
e		1.27			0.050	
F	0.33		0.53	0.013		0.021
F1	0.66		0.81	0.026		0.032
F2	0.64			0.025		
G			0.10			0.004
M	1.07		1.42	0.042		0.056
M1	1.07		1.22	0.042		0.048
	Number of Pins					
N	84					
ND	21					
NE	21					

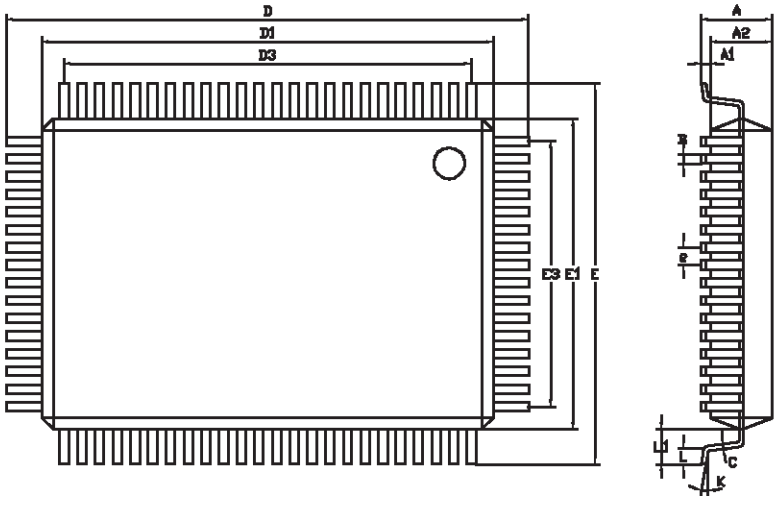
84-PIN CERAMIC LEADED CHIP CARRIER PACKAGE



CLCC684W

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A		4.01			0.158	
A1	1.14	1.78	2.41	0.045	0.070	0.095
A2		0.20			0.008	
B	0.33	0.43	0.53	0.013	0.017	0.021
B1	0.66	0.76	0.81	0.026	0.030	0.032
E	29.72	30.23	30.73	1.170	1.190	1.210
E1	28.65	29.21	29.77	1.128	1.150	1.172
E3		25.40			1.000	
e		1.27			0.050	
G	17.78	18.03	18.29	0.700	0.710	0.720
G2		1.09			0.043	
M1		1.02			0.040	
m1		0.51			0.020	
Ø		12.70			0.500	
	Number of Pins					
N	84					
ND	21					
NE	21					

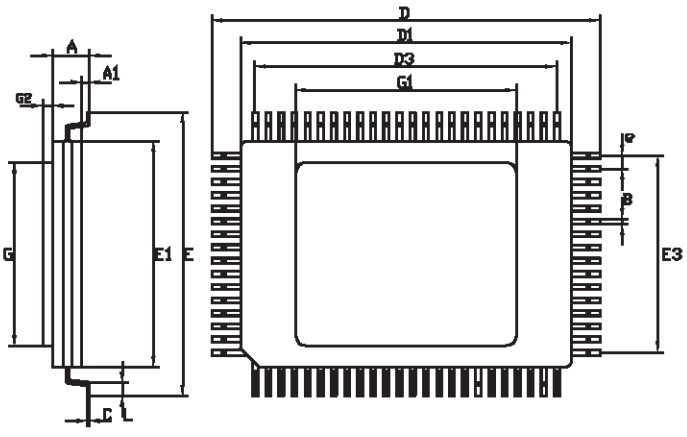
80-PIN PLASTIC QUAD FLAT PACKAGE



PQFP080

Dim	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.40			0.134
A1	0.25			0.010		
A2	2.55	2.80	3.05	0.100	0.110	0.120
B	0.30		0.45	0.012		0.018
C	0.13		0.23	0.005		0.009
D	22.95	23.20	23.45	0.904	0.913	0.923
D1	19.90	20.00	20.10	0.783	0.787	0.791
D3		18.40			0.724	
E	16.95	17.20	17.45	0.667	0.677	0.687
E1	13.90	14.00	14.10	0.547	0.551	0.555
E3		12.00			0.472	
e		0.80			0.031	
K	0°		7°			
L	0.65	0.80	0.95	0.026	0.031	0.037
L1		1.60			0.063	
	Number of Pins					
N	80		ND	24	NE	16

80-PIN CERAMIC QUAD FLAT PACKAGE



CQFP080

Dim	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.24			0.128
A1		0.20			0.008	
B	0.30	0.35	0.45	0.012	0.014	0.018
C	0.13	0.15	0.23	0.005	0.006	0.009
D	23.35	23.90	24.45	0.919	0.941	0.963
D1	19.57	20.00	20.43	0.770	0.787	0.804
D3		18.40			0.724	
E	17.35	17.90	18.45	0.683	0.705	0.726
E1	13.61	14.00	14.39	0.536	0.551	0.567
E3		12.00			0.472	
e		0.80			0.031	
G	13.75	14.00	14.25	0.541	0.551	0.561
G1	19.75	20.00	20.25	0.778	0.787	0.797
G2		1.06			0.042	
L	0.35	0.80		0.014	0.031	
	Number of Pins					
N	80					

11.2 ORDERING INFORMATION

Part Number	Program Memory (Bytes)	RAM (Bytes)	Watchdog Option	Temperature Range	Package
ST90135M4Q6	16K ROM	512	HW/SW	-40°C +85°C 0°C +70°C	PQFP80
ST90135M5Q6	24K ROM	768			
ST90135M6Q6	32K ROM	1K			
ST90158M7Q6	48K ROM	1.5K			
ST90158M9Q6	64K ROM	2K			
ST90158P9C6			SW	-40°C +85°C	PLCC84
ST90E158M9G0	64K EPROM		HW/SW	+ 25°C	CQFP80-W
ST90E158P9L0			SW		CLCC84-W
ST90T158M9Q6	64K OTP		HW/SW	-40°C +85°C	PQFP80
ST90T158P9C6			SW		PLCC84
ST90R158Q6	ROMless	2K	HW/SW	-40°C +85°C	PQFP80

Notes: