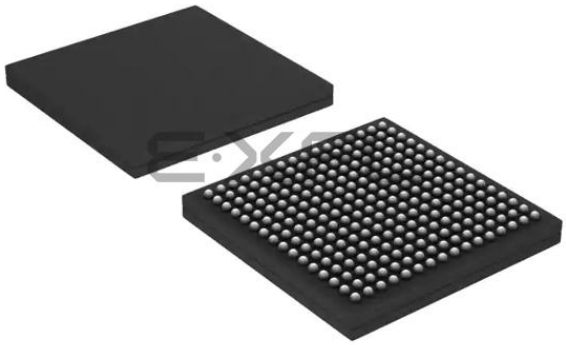


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Not For New Designs
Core Processor	Coldfire V2
Core Size	32-Bit Single-Core
Speed	80MHz
Connectivity	CANbus, EBI/EMI, Ethernet, I²C, SPI, UART/USART
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	142
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	64K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	256-LBGA
Supplier Device Package	256-MAPBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcf5280cvm80

Appendix A
Register Memory Map

Appendix B
Revision History

ET with {<ea> = (d16,PC)}	equals ET with {<ea> = (d16,An)}
ET with {<ea> = (d8,PC,Xi*SF)}	equals ET with {<ea> = (d8,An,Xi*SF)}

The nomenclature xxx.wl refers to both forms of absolute addressing, xxx.w and xxx.l.

Table 2-12. MOVE Byte and Word Execution Times

Source	Destination						
	Rx	(Ax)	(Ax)+	-(Ax)	(d16,Ax)	(d8,Ax,Xi*SF)	xxx.wl
Dy	1(0/0)	1(0/1)	1(0/1)	1(0/1)	1(0/1)	2(0/1)	1(0/1)
Ay	1(0/0)	1(0/1)	1(0/1)	1(0/1)	1(0/1)	2(0/1)	1(0/1)
(Ay)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	3(1/1)	4(1/1))	3(1/1)
(Ay)+	3(1/0)	3(1/1)	3(1/1)	3(1/1)	3(1/1)	4(1/1))	3(1/1)
-(Ay)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	3(1/1)	4(1/1))	3(1/1)
(d16,Ay)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	3(1/1)	—	—
(d8,Ay,Xi*SF)	4(1/0)	4(1/1)	4(1/1)	4(1/1)	—	—	—
xxx.w	3(1/0)	3(1/1)	3(1/1)	3(1/1)	—	—	—
xxx.l	3(1/0)	3(1/1)	3(1/1)	3(1/1)	—	—	—
(d16,PC)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	3(1/1)	—	—
(d8,PC,Xi*SF)	4(1/0)	4(1/1)	4(1/1)	4(1/1))	—	—	—
#xxx	1(0/0)	3(0/1)	3(0/1)	3(0/1)	—	—	—

Table 2-13. MOVE Long Execution Times

Source	Destination						
	Rx	(Ax)	(Ax)+	-(Ax)	(d16,Ax)	(d8,Ax,Xi*SF)	xxx.wl
Dy	1(0/0)	1(0/1)	1(0/1)	1(0/1)	1(0/1)	2(0/1)	1(0/1)
Ay	1(0/0)	1(0/1)	1(0/1)	1(0/1)	1(0/1)	2(0/1)	1(0/1)
(Ay)	2(1/0)	2(1/1)	2(1/1)	2(1/1)	2(1/1)	3(1/1)	2(1/1)
(Ay)+	2(1/0)	2(1/1)	2(1/1)	2(1/1)	2(1/1)	3(1/1)	2(1/1)
-(Ay)	2(1/0)	2(1/1)	2(1/1)	2(1/1)	2(1/1)	3(1/1)	2(1/1)
(d16,Ay)	2(1/0)	2(1/1)	2(1/1)	2(1/1)	2(1/1)	—	—
(d8,Ay,Xi*SF)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	—	—	—
xxx.w	2(1/0)	2(1/1)	2(1/1)	2(1/1)	—	—	—
xxx.l	2(1/0)	2(1/1)	2(1/1)	2(1/1)	—	—	—
(d16,PC)	2(1/0)	2(1/1)	2(1/1)	2(1/1)	2(1/1)	—	—
(d8,PC,Xi*SF)	3(1/0)	3(1/1)	3(1/1)	3(1/1)	—	—	—
#xxx	1(0/0)	2(0/1)	2(0/1)	2(0/1)	—	—	—

Chapter 5
Static RAM (SRAM)

5.1 SRAM Features

- One 64-Kbyte SRAM
- Single-cycle access
- Physically located on processor's high-speed local bus
- Memory location programmable on any 0-modulo-64 Kbyte address
- Byte, word, longword address capabilities

5.2 SRAM Operation

The SRAM module provides a general-purpose memory block that the ColdFire processor can access in a single cycle. The location of the memory block can be specified to any 0-modulo-64K address within the 4-GByte address space. The memory is ideal for storing critical code or data structures or for use as the system stack. Because the SRAM module is physically connected to the processor's high-speed local bus, it can service processor-initiated access or memory-referencing commands from the debug module.

Depending on configuration information, instruction fetches may be sent to both the cache and the SRAM block simultaneously. If the reference is mapped into the region defined by the SRAM, the SRAM provides the data back to the processor, and the cache data discarded. Accesses from the SRAM module are not cached.

The SRAM is dual-ported to provide DMA access. The SRAM is partitioned into two physical memory arrays to allow simultaneous access to both arrays by the processor core and another bus master. See Chapter 8, “System Control Module (SCM)” for more information.

5.3 SRAM Programming Model

The SRAM programming model includes a description of the SRAM base address register (RAMBAR), SRAM initialization, and power management.

5.3.1 SRAM Base Address Register (RAMBAR)

The configuration information in the SRAM base address register (RAMBAR) controls the operation of the SRAM module.

- The RAMBAR holds the base address of the SRAM. The MOVEC instruction provides write-only access to this register.
- The RAMBAR can be read or written from the debug module in a similar manner.
- All undefined bits in the register are reserved. These bits are ignored during writes to the RAMBAR, and return zeroes when read from the debug module.
- The RAMBAR valid bit is cleared by reset, disabling the SRAM module. All other bits are unaffected.

12.4.1 Chip Select Module Registers

The chip select module is programmed through the chip select address registers (CSAR0–CSAR6), chip select mask registers (CSMR0–CSMR6), and the chip select control registers (CSCR0–CSCR6).

12.4.1.1 Chip Select Address Registers (CSAR0–CSAR6)

The CSARs, Figure 12-2, specify the chip select base addresses.

	15 … 0
Field	BA
Reset	Uninitialized
R/W	R/W
Address	0x080 (CSAR0); 0x08C (CSAR1); 0x098 (CSAR2); 0x0A4 (CSAR3); 0x0B0 (CSAR4); 0x0BC (CSAR5); 0x0C8 (CSAR6)

Figure 12-2. Chip Select Address Registers (CSAR*n*)

Table 12-6 describes CSAR[BA].

Table 12-6. CSAR*n* Field Description

Bits	Name	Description
15–0	BA	Base address. Defines the base address for memory dedicated to chip select $\overline{CS}$[6:0]. BA is compared to bits 31–16 on the internal address bus to determine if chip select memory is being accessed.

12.4.1.2 Chip Select Mask Registers (CSMR0–CSMR6)

The CSMRs, Figure 12-3, are used to specify the address mask and allowable access types for the respective chip selects.

	31 … 16	15 … 9	8	7	6	5	4	3	2	1	0
Field	BAM	—	WP	—	AM	C/I	SC	SD	UC	UD	V
Reset	Unitialized										0
R/W	R/W										
Addr	0x084 (CSMR0); 0x090 (CSMR1); 0x09C (CSMR2); 0x0A8 (CSMR3); 0x0B4 (CSMR4); 0x0C0 (CSMR5); 0x0CC (CSMR6)										

Figure 12-3. Chip Select Mask Registers (CSMR*n*)

Table 12-7 describes CSMR fields.

Table 14-3. MCF5282 Signals and Pin Numbers Sorted by Function (continued)

MAPBGA Pin	Pin Functions			Description	Primary I/O	Internal Pull-up[1]
	Primary[2]	Secondary	Tertiary			
J16	DTIN1	PTD3	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	U1/U0 Request to Send	I/O	—
J15	DTOUT1	PTD2	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	U1/U0 Request to Send	I/O	—
J14	DTIN0	PTD1	$\overline{\text{UCTS1}}$/$\overline{\text{UCTS0}}$	U1/U0 Clear to Send	I/O	—
J13	DTOUT0	PTD0	$\overline{\text{UCTS1}}$/$\overline{\text{UCTS0}}$	U1/U0 Clear to Send	I/O	—
Note: The below two pins are for the MCF5280, MCF5281, and MCF5282 only.						
C10	EMDIO	PAS5	URXD2	U2 receive data	I/O	—
B10	EMDC	PAS4	UTXD2	U2 transmit data	I/O	—
Note: The below two pins are for the MCF5214 and MCF5216 only.						
C10	PAS5	URXD2	—	U2 receive data	I/O	—
B10	NC	—	—	No connect	I/O	—
General Purpose Timers						
T13:R13:P13:N13	GPTA[3:0]	PTA[3:0]	—	Timer A IC/OC/PAI	I/O	Yes
T12:R12:P12:N12	GPTB[3:0]	PTB[3:0]	—	Timer B IC/OC/PAI	I/O	Yes
N14	SIZ1	PE3	SYNCA	Timer A synchronization input	I/O	Yes[3]
M16	SIZ0	PE2	SYNCB	Timer B synchronization input	I/O	Yes[4]
M15	$\overline{\text{TS}}$	PE1	SYNCA	Timer A synchronization input	I/O	Yes
M14	$\overline{\text{TIP}}$	PE0	SYNCB	Timer B synchronization input	I/O	Yes
DMA Timers						
K16	DTIN3	PTC3	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	Timer 3 in	I/O	—
K15	DTOUT3	PTC2	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	Timer 3 out	I/O	—
K14	DTIN2	PTC1	$\overline{\text{UCTS1}}$/$\overline{\text{UCTS0}}$	Timer 2 in	I/O	—
K13	DTOUT2	PTC0	$\overline{\text{UCTS1}}$/$\overline{\text{UCTS0}}$	Timer 2 out	I/O	—
J16	DTIN1	PTD3	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	Timer 1 in	I/O	—
J15	DTOUT1	PTD2	$\overline{\text{URTS1}}$/$\overline{\text{URTS0}}$	Timer 1 out	I/O	—

14.2.6.14 Receive Error (ERXER)

ERXER is an input signal which when asserted along with ERXDV signals that the PHY has detected an error in the current frame. When ERXDV is not asserted ERXER has no effect, and applies to MII mode operation.

These pins can also be configured as GPIO PEL0.

14.2.7 Queued Serial Peripheral Interface (QSPI) Signals

14.2.7.1 QSPI Synchronous Serial Output (QSPI_DOUT)

The QSPI_DOUT output provides the serial data from the QSPI and can be programmed to be driven on the rising or falling edge of QSPICLK. Each byte is sent msb first.

This pin can also be configured as GPIO PQS0.

14.2.7.2 QSPI Synchronous Serial Data Input (QSPI_DIN)

The QSPI_DIN input provides the serial data to the QSPI and can be programmed to be sampled on the rising or falling edge of QSPICLK. Each byte is written to RAM lsb first.

This pin can also be configured as GPIO PQS1.

14.2.7.3 QSPI Serial Clock (QSPI_CLK)

The QSPI serial clock (QSPI_CLK) provides the serial clock from the QSPI. The polarity and phase of QSPI_CLK are programmable. The output frequency is programmed according to the following formula, in which n can be any value between 2 and 255: QSPI_CLK = CLKOUT/(2n).

This pin can also be configured as GPIO PQS2.

14.2.7.4 QSPI Chip Selects (QSPI_CS[3:0])

The synchronous peripheral chip selects (QSPI_CS[3:0]) outputs provide QSPI peripheral chip selects that can be programmed to be active high or low.

This pin can also be configured as GPIO PQS[6:3].

14.2.8 FlexCAN Signals

14.2.8.1 FlexCAN Transmit (CANTX)

Controller Area Network Transmit data output.

This pin can also be configured as GPIO PAS2.

14.2.8.2 FlexCAN Receive (CANRX)

Controller Area Network Transmit data input.

This pin can also be configured as GPIO PAS3.

17.4.1 MIB Block Counters Memory Map

The MIB counters memory map (Table 17-4) defines the locations in the MIB RAM space where hardware-maintained counters reside. The counters are divided into two groups:

- RMON counters include the Ethernet statistics counters defined in RFC 1757
- A counter is included to count truncated frames since only frame lengths up to 2047 bytes are supported

The transmit and receive RMON counters are independent, which ensures accurate network statistics when operating in full duplex mode.

The included IEEE counters support the mandatory and recommended counter packages defined in Section 5 of ANSI/IEEE Std. 802.3 (1998 edition). The FEC supports IEEE Basic Package objects, but these do not require counters in the MIB block. In addition, some of the recommended package objects supported do not require MIB counters. Counters for transmit and receive full duplex flow control frames are also included.

Table 17-4. MIB Counters Memory Map

IPSBAR Offset	Register
0x1200	Count of frames not counted correctly (RMON_T_DROP)
0x1204	RMON Tx packet count (RMON_T_PACKETS)
0x1208	RMON Tx broadcast packets (RMON_T_BC_PKT)
0x120C	RMON Tx multicast packets (RMON_T_MC_PKT)
0x1210	RMON Tx packets with CRC/align error (RMON_T_CRC_ALIGN)
0x1214	RMON Tx packets < 64 bytes, good CRC (RMON_T_UNDERSIZE)
0x1218	RMON Tx packets > MAX_FL bytes, good CRC (RMON_T_OVERSIZE)
0x121C	RMON Tx packets < 64 bytes, bad CRC (RMON_T_FRAG)
0x1220	RMON Tx packets > MAX_FL bytes, bad CRC (RMON_T_JAB)
0x1224	RMON Tx collision count (RMON_T_COL)
0x1228	RMON Tx 64 byte packets (RMON_T_P64)
0x122C	RMON Tx 65 to 127 byte packets (RMON_T_P65TO127)
0x1230	RMON Tx 128 to 255 byte packets (RMON_T_P128TO255)
0x1234	RMON Tx 256 to 511 byte packets (RMON_T_P256TO511)
0x1238	RMON Tx 512 to 1023 byte packets (RMON_T_P512TO1023)
0x123C	RMON Tx 1024 to 2047 byte packets (RMON_T_P1024TO2047)
0x1240	RMON Tx packets with > 2048 bytes (RMON_T_P_GTE2048)
0x1244	RMON Tx Octets (RMON_T_OCTETS)
0x1248	Count of transmitted frames not counted correctly (IEEE_T_DROP)
0x124C	Frames transmitted OK (IEEE_T_FRAME_OK)
0x1250	Frames transmitted with single collision (IEEE_T_1COL)

Table 18-4. WMR Field Descriptions

Bit(s)	Name	Description
15–0	WM	Watchdog modulus. Contains the modulus that is reloaded into the watchdog counter by a service sequence. Once written, the WM[15:0] field is not affected by further writes except in halted mode. Writing to WMR immediately loads the new modulus value into the watchdog counter. The new value is also used at the next and all subsequent reloads. Reading WMR returns the value in the modulus register. Reset initializes the WM[15:0] field to 0xFFFF. Note: The prescaler counter is reset anytime a new value is loaded into the watchdog counter and also during reset.

18.5.2.3 Watchdog Count Register (WCNTR)

	15	14	13	12	11	10	9	8
Field	WC15	WC14	WC13	WC12	WC11	WC10	WC9	WC8
Reset	1111_1111							
R/W	R							

	7	6	5	4	3	2	1	0
Field	WC7	WC6	WC5	WC4	WC3	WC2	WC1	WC0
Reset	1111_1111							
R/W	R							
Address	IPSBAR + x0014_0004, 0x0014_0005							

Figure 18-4. Watchdog Count Register (WCNTR)

Table 18-5. WCNTR Field Descriptions

Bit(s)	Name	Description
15–0	WC	Watchdog count field. Reflects the current value in the watchdog counter. Reading the 16-bit WCNTR with two 8-bit reads is not guaranteed to return a coherent value. Writing to WCNTR has no effect, and write cycles are terminated normally.

18.5.2.4 Watchdog Service Register (WSR)

When the watchdog timer is enabled, writing 0x5555 and then 0xAAAA to WSR before the watchdog counter times out prevents a reset. If WSR is not serviced before the timeout, the watchdog timer sends a signal to the reset controller module that sets the RSR[WDR] bit and asserts a system reset.

Both writes must occur in the order listed before the timeout, but any number of instructions can be executed between the two writes. However, writing any value other than 0x5555 or 0xAAAA to WSR resets the servicing sequence, requiring both values to be written to keep the watchdog timer from causing a reset.

To configure the UART for DMA requests:

1. Initialize the DMAREQC in the SCM to map the desired UART DMA requests to the desired DMA channels. For example, setting DMAREQC[7:4] to 1000 maps UART0 receive DMA requests to DMA channel 1, setting DMAREQC[11:8] to 1101 maps UART1 transmit DMA requests to DMA channel 2, and so on. It is possible to independently map transmit-based and receive-based UART DMA requests in the DMAREQC.
2. Disable interrupts using the UIMR register. The appropriate UIMR bits must be cleared so that interrupt requests are disabled for those conditions for which a DMA request is desired. For example, to generate transmit DMA requests from UART1, UIMR1[TXRDY] should be cleared. This prevents TXRDY from generating an interrupt request while a transmit DMA request is generated.
3. Enable DMA access to the UART*n* registers by setting the corresponding PACR register in the SCM for read/write in supervisor and user modes.
4. Enable DMA access to SRAM by setting the SPV bit in the core RAMBAR, and the BDE bit in the SCM RAMBAR
5. Initialize the DMA channel. The DMA should be configured for cycle steal mode and a source and destination size of one byte. This causes a single byte to be transferred for each UART DMA request. Set the disable request bit (DCR*n*[D_REQ] to disable external requests when the BCR reaches zero.
6. For a transmit process:
   — Set the DMA SAR register to the address of the source data
   — Set DCR*n*[SINC] to increment the source pointer
   — Set DAR to the address if the UART transmit buffer (UTB)
   — Clear DCR*n*[DINC]
   — Set BCR to the number of bytes to transmit.
7. For a receive process:
   — Set the DMA SAR register to the address of the UART receive buffer (URB)
   — Clear DCR*n*[SINC]
   — Set DAR to the address of the source data
   — Set DCR*n*[DINC] to increment the destination pointer
   — Set BCR to the number of bytes to transmit.
8. Start the data transfer by setting DCR*n*[EEXT], which enables the UART channel to issue DMA requests.

Table 23-14 shows the DMA requests.

Table 23-14. UART DMA Requests

Register	Bit	DMA Request
UISR*n*	1	Receive DMA request
UISR*n*	0	Transmit DMA request

Table 25-17. ESTAT Field Descriptions (continued)

Bits	Name	Description
8	RXWARN	Receiver error status flag. The RXWARN status flag reflects the status of the FlexCAN receive error counter. 0 Receive error counter < 96. 1 Receive error counter ≥ 96.
7	IDLE	Idle status. The IDLE bit indicates when there is activity on the CAN bus. 0 The CAN bus is not idle. 1 The CAN bus is idle.
6	TX/RX	Transmit/receive status. The TX/RX bit indicates when the FlexCAN module is transmitting or receiving a message. TX/RX has no meaning when IDLE = 1. 0 The FlexCAN is receiving a message if IDLE = 0. 1 The FlexCAN is transmitting a message if IDLE = 0.
5–4	FCS	Fault confinement state. The FCS[1:0] field describes the state of the FlexCAN. If the SOFTRST bit in CANMCR is asserted while the FlexCAN is in the bus off state, the error and status register is reset, including FCS[1:0]. However, as soon as the FlexCAN exits reset, FCS[1:0] bits will again reflect the bus off state. Refer to Section 25.5.11, “FlexCAN Receive Error Counter (RXECTR)” for more information on entry into and exit from the various fault confinement states. 00 Error active 01 Error passive 1X Reserved
3	—	Reserved, should be cleared.
2	BOFFINT	Bus off interrupt. The BOFFINT bit is used to request an interrupt when the FlexCAN enters the bus off state. To clear this bit, first read it as a one, then write a one. Writing zero has no effect. 0 No bus off interrupt requested. 1 When the FlexCAN state changes to bus off, this bit is set, and if the BOFFMSK bit in CANCTRL0 is set, an interrupt request is generated. This interrupt is not requested after reset.
1	ERRINT	Error interrupt. The ERRINT bit is used to request an interrupt when the FlexCAN detects a transmit or receive error. To clear this bit, first read it as a one, then write a one. Writing zero has no effect. 0 No error interrupt request. 1 If an event which causes one of the error bits in the error and status register to be set occurs, the error interrupt bit is set. If the ERRMSK bit in CANCTRL0 is set, an interrupt request is generated.
0	WAKEINT	Wake interrupt. The WAKEINT bit indicates that bus activity has been detected while the FlexCAN module is in low-power stop mode. To clear this bit, first read it as a one, then write a one. Writing zero has no effect. 0 No wake interrupt requested. 1 When the FlexCAN is in low-power stop mode and a recessive to dominant transition is detected on the CAN bus, this bit is set. If the WAKEMSK bit is set in CANMCR, an interrupt request is generated.

25.5.9 Interrupt Mask Register (IMASK)

IMASK contains one interrupt mask bit per buffer. It enables the CPU to determine which buffer will generate an interrupt after a successful transmission/reception (that is, when the corresponding IFLAG bit is set).

Table 26-1. Ports External Signals

Primary Function (Pin Name)[1]	GPIO (Default Function)	Alternate Function 1	Alternate Function 2	Description
D[31:0][2]	PA, PB, PC, PD	—	—	External data bus / Ports A, B, C, D
$\overline{OE}$[1]	PE[7]	—	—	Output enable for external reads / Port E[7]
$\overline{TA}$[1]	PE[6]	—	—	Transfer acknowledge for external data transfer / Port E[6]
$\overline{TEA}$[1]	PE[5]	—	—	Transfer error acknowledge for external data transfer / Port E[5]
R/$\overline{W}$[1]	PE[4]	—	—	Read/Write indication for external data transfer / Port E[4]
SIZ1[1]	PE[3]	SYNCA	—	Size of the external data transfer / Port E[3] / Timer A sync
SIZ0[1]	PE[2]	SYNCB	—	Size of the external data transfer / Port E[3:2] / Timer B sync
$\overline{TS}$[1]	PE[1]	SYNCA	—	Transfer start indication for external data transfer / Port E[1] / Timer A sync
$\overline{TIP}$[1]	PE[0]	SYNCB	—	Transfer in progress indication for external data transfer / Port E[0] / Timer B sync
A[23:21][1]	PF[7:5]	$\overline{CS}$[6:4]	—	External address bus [23:21] / Port F[7:5] / Chip selects 6–4
A[20:0][1]	PF[4:0], PG, PH	—	—	External address bus [20:0] / Ports F[4:0], G, H
$\overline{BS}$[3:0][1]	PJ[7:4]	—	—	Byte strobes for external data transfer / Port J[7:4] / SDRAM column address strobes
$\overline{CS}$[3:0][1]	PJ[3:0]	—	—	Chip selects 3 - 0 / Port J[3:0]
DDATA[3:0]	PDD[7:4]	—	—	Debug data / Port DD[7:4]
PST[3:0]	PDD[7:4]	—	—	Processor status / Port DD[3:0]
CANRX	PAS[3]	URXD2	—	FlexCAN receive data / Port AS[3] / URXD2
CANTX	PAS[2]	UTXD2	—	FlexCAN transmit data / Port AS[2] / UTXD2
SDA	PAS[1]	URXD2	—	I^2C serial data / Port AS[1] / URXD2
SCL	PAS[0]	UTXD2	—	I^2C serial clock / Port AS[0] / UTXD2
IRQ[7:1][3]	PNQ[7:1]	—	—	Edge Port external interrupt pins / Port NQ[7:1]
AN[56:55:53:52][2]	PQA [4:3:1:0]	ETRIG[2:1], MA[1:0]	—	QADC analog inputs / Port QA[4:3:1:0] / external triggers / external multiplex control
AN[3:0] [2]	PQB[3:0]	ANZ, ANY, ANX, ANW	—	QADC analog inputs / Port QB[3:0] / multiplexed analog inputs
QSPI_CS [3:0]	PQS[6:3]	—	—	QSPI synchronous peripheral chip selects / Port QS[3:6]
QSPI_CLK	PQS[2]	—	—	QSPI serial clock / Port QS[2]
QSPI_DIN	PQS[1]	—	—	QSPI serial data input / Port QS[1]
QSPI_DOUT	PQS[0]	—	—	QSPI serial data output / Port QS[0]
$\overline{SRAS}$	PSD[5]	—	—	SDRAM synchronous row address strobe / Port SD[5]

NOTE

When Flash security is enabled, the chip will boot in single chip mode regardless of the external reset configuration.

Table 27-9. Chip Configuration Mode Selection[1]

Chip Configuration Mode	CCR Register MODE Field		
	MODE2	MODE1	MODE0
Master mode	D26 driven high	D17 driven high	D16 driven high
Single-chip mode	D26 driven high	D17 driven high	D16 driven low
Reserved	D26 driven high	D17 driven low	D16 driven high
Reserved	D26 driven low	D17 don't care	D16 don't care

1 Modifying the default configurations is possible only if the external $\overline{\text{RCON}}$ pin is asserted low.

During reset, certain module configurations depend on whether emulation mode is active as determined by the state of the internal emulation signal.

27.6.3 Boot Device Selection

During reset configuration, the $\overline{\text{CS0}}$ chip select pin is optionally configured to select an external boot device. In this case, the V (valid) bit in the CSMR0 register is ignored, and $\overline{\text{CS0}}$ is enabled after reset. $\overline{\text{CS0}}$ is asserted for the initial boot fetch accessed from address 0x0000_0000 for the Stack Pointer and address 0x0000_0004 for the program counter (PC). It is assumed that the reset vector loaded from address 0x0000_0004 causes the CPU to start executing from external memory space decoded by $\overline{\text{CS0}}$.

27.6.4 Output Pad Strength Configuration

Output pad strength is determined during reset configuration as shown in Table 27-10. Once reset is exited, the output pad strength configuration can be changed by programming the LOAD bit of the chip configuration register.

Table 27-10. Output Pad Driver Strength Selection[1]

Optional Pin Function Selection	CCR Register LOAD Bit
Output pads configured for partial strength	D21 driven low
Output pads configured for full strength	D21 driven high

1 Modifying the default configurations is possible only if the external $\overline{\text{RCON}}$ pin is asserted low.

27.6.5 Clock Mode Selection

The clock mode is selected during reset by the CLKMOD pins and reflected in the PLLMODE, PLLSEL, and PLLREF bits of SYNSR. After reset is exited, the clock mode cannot be changed.

Table 27-11 summarizes clock mode selection during reset configuration.

[1] All channels not listed are reserved or unimplemented and return undefined results.

28.6.8 Result Registers

The result word table is a 64 half-word (128 byte) long by 10-bit wide RAM. An entry is written by the QADC after completing an analog conversion specified by the corresponding CCW table entry.

28.6.8.1 Right-Justified Unsigned Result Register (RJURR)

	15 … 10	9 … 8
Field	—	RESULT
Reset	0000_00	Undefined
R/W:	R	R/W

	7 … 0
Field	RESULT
Reset	Undefined
R/W:	R/W
Address	IPSBAR + 0x19_0280, 0x19_02fe

Figure 28-15. Right-Justified Unsigned Result Register (RJURR)

Table 28-18. RJURR Field Descriptions

Bit(s)	Name	Description
15–10	—	Reserved, should be cleared.
9–0	RESULT	The conversion result is unsigned, right-justified data.

28.6.8.2 Left-Justified Signed Result Register (LJSRR)

	15	14 … 8
Field	S	RESULT
Reset	Undefined	
R/W:	R/W	

	7 … 0
Field	RESULT
Reset	Undefined
R/W:	R/W
Address	IPSBAR + 0x19_0300, 0x19_037e

Figure 28-16. Left-Justified Signed Result Register (LJSRR)

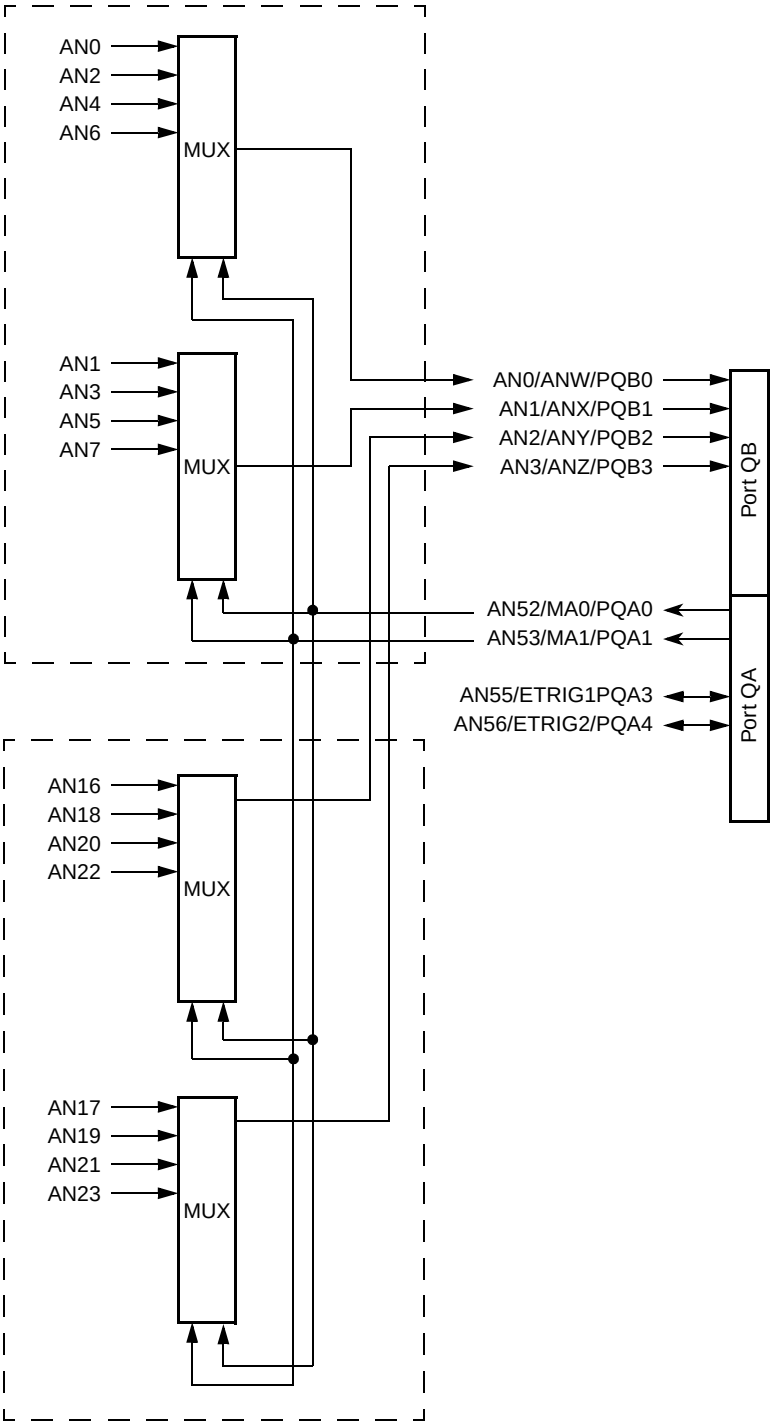
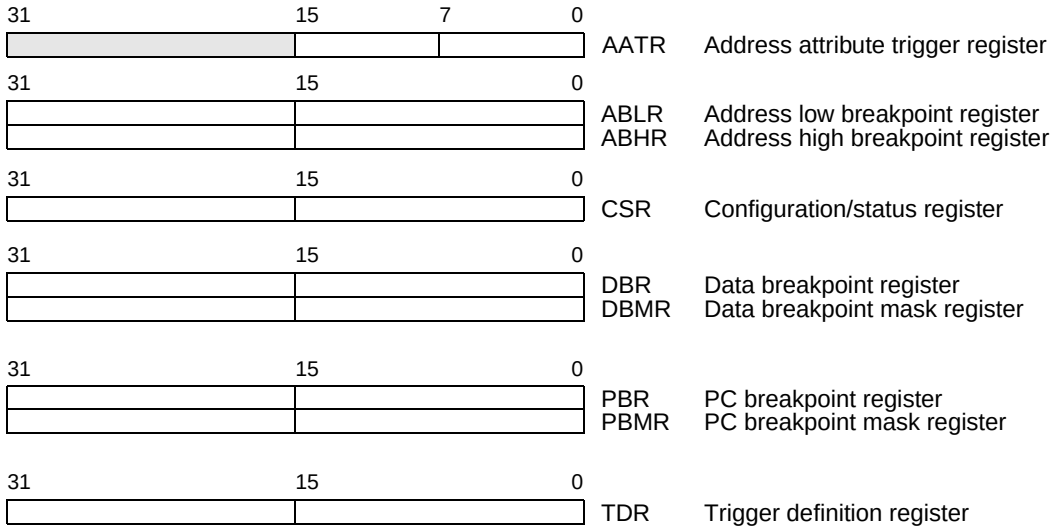
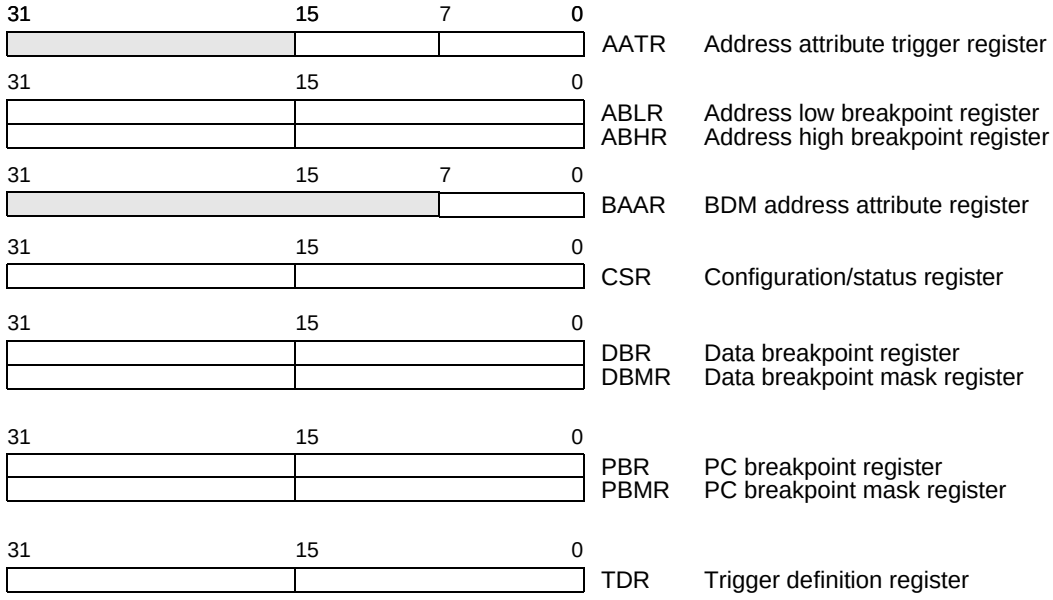


Figure 28-18. External Multiplexing Configuration

When externally multiplexed mode is selected, the QADC automatically drives the MA output signals from the channel number in each CCW. The QADC also converts the proper input channel (ANW, ANX, ANY, and ANZ) by interpreting the CCW channel number. As a result, up to 16 externally multiplexed channels appear to the conversion queues as directly connected signals. User software simply puts the channel number of externally multiplexed channels into CCWs.



Note: Each debug register is accessed as a 32-bit register; shaded fields above are not used (don't care). All debug control registers are writable from the external development system or the CPU via the WDEBUG instruction. CSR is write-only from the programming model. It can be read or written through the BDM port using the RDMREG and WDMREG commands.



Note: Each debug register is accessed as a 32-bit register; shaded fields above are not used (don't care). All debug control registers are writable from the external development system or the CPU via the WDEBUG instruction.
CSR is write-only from the programming model. It can be read or written through the BDM port using the RDMREG and WDMREG commands.

Figure 30-4. Debug Programming Model

These registers are accessed through the BDM port by the commands, WDMREG and RDMREG, described in Section 30.5.3.3, "Command Set Descriptions." These commands contain a 5-bit field, DRc, that specifies the register, as shown in Table 30-3.

Command Sequence:

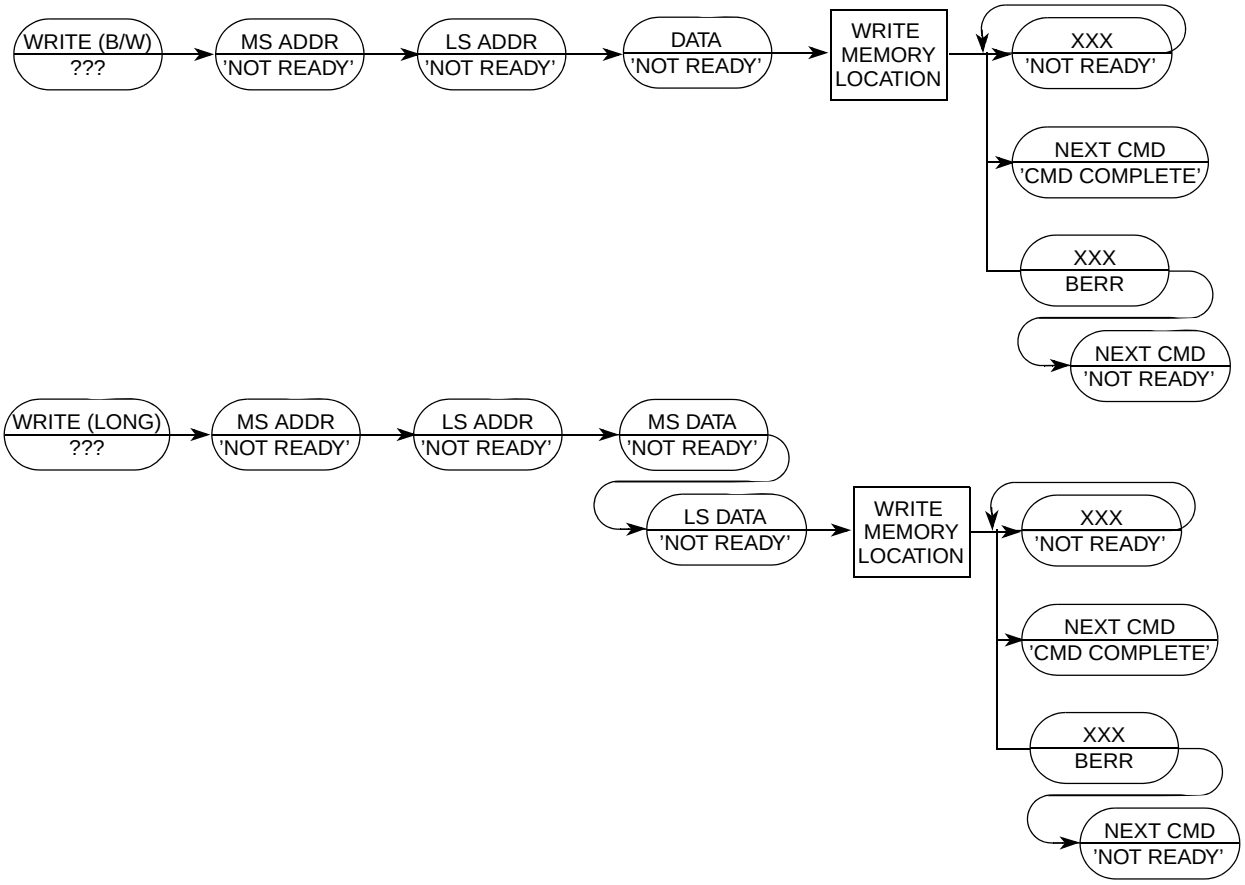


Figure 30-24. WRITE Command Sequence

Operand Data	This two-operand instruction requires a longword absolute address that specifies a location to which the data operand is to be written. Byte data is sent as a 16-bit word, justified in the LSB; 16- and 32-bit operands are sent as 16 and 32 bits, respectively
Result Data	Command complete status is indicated by returning 0xFFFF (with S cleared) when the register write is complete. A value of 0x0001 (with S set) is returned if a bus error occurs.

30.5.3.3.5 Dump Memory Block (DUMP)

DUMP is used with the READ command to access large blocks of memory. An initial READ is executed to set up the starting address of the block and to retrieve the first result. If an initial READ is not executed before the first DUMP, an illegal command response is returned. The DUMP command retrieves subsequent operands. The initial address is incremented by the operand size (1, 2, or 4) and saved in a temporary register. Subsequent DUMP commands use this address, perform the memory read, increment it by the current operand size, and store the updated address in the temporary register.

For all types of exception processing, the PST = 0xC value is driven at all times, unless the PST output is needed for one of the optional marker values or for the taken branch indicator (0x5).

Table 30-23 shows the PST/DDATA specification for multiply-accumulate instructions.

Table 30-23. PST/DDATA Specification for MAC Instructions

Instruction	Operand Syntax	PST/DDATA
mac.l	Ry,Rx,Accx	PST = 0x1
mac.l	RyRx,<ea>,Rw,Accx	PST = 0x1, {PST = 0xB, DD = source operand}
mac.w	Ry,Rx,Accx	PST = 0x1
mac.w	Ry,Rx,<ea>,Rw,Accx	PST = 0x1, {PST = 0xB, DD = source operand}
move.l	<ea>y,Accx	PST = 0x1
move.l	Accy,Accx	PST = 0x1
move.l	<ea>y,MACR	PST = 0x1
move.l	<ea>y,MASK	PST = 0x1
move.l	<ea>y,Accext01	PST = 0x1
move.l	<ea>y,Accext23	PST = 0x1
move.l	Accy,Rx	PST = 0x1
move.l	MACSR,CCR	PST = 0x1
move.l	MACSR,Rx	PST = 0x1
move.l	MASK,Rx	PST = 0x1
move.l	Accext01,Rx	PST = 0x1
move.l	Accext23,Rx	PST = 0x1
msac.l	Ry,Rx,Accx	PST = 0x1
msac.l	Ry,Rx,<ea>,Rw,Accx	PST = 0x1, {PST = 0xB, DD = source operand}
msac.w	Ry,Rx,Accx	PST = 0x1
msac.w	Ry,Rx,<ea>,Rw,Accx	PST = 0x1, {PST = 0xB, DD = source operand}

30.7.2 Supervisor Instruction Set

The supervisor instruction set has complete access to the user mode instructions plus the opcodes shown below. The PST/DDATA specification for these opcodes is shown in Table 30-24.

Table 30-24. PST/DDATA Specification for Supervisor-Mode Instructions

Instruction	Operand Syntax	PST/DDATA
cpushl		PST = 0x1
halt		PST = 0x1, PST = 0xF
move.w	SR,Dx	PST = 0x1
move.w	{Dy,#imm},SR	PST = 0x1, {PST = 0x3}
movec	Ry,Rc	PST = 0x1

D

E