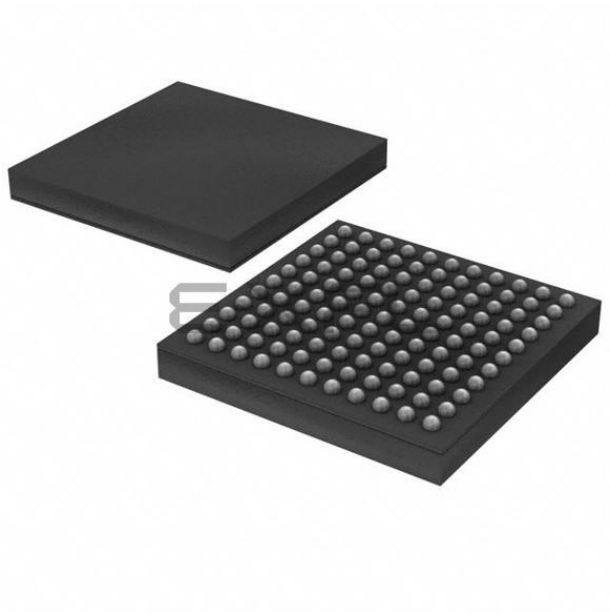


Welcome to E-XFL.COM

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	Discontinued at Digi-Key
Core Processor	ARM® Cortex®-M3
Core Size	32-Bit Single-Core
Speed	48MHz
Connectivity	EBI/EMI, I²C, IrDA, SmartCard, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, LCD, POR, PWM, WDT
Number of I/O	93
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.85V ~ 3.8V
Data Converters	A/D 8x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	120-VFBGA
Supplier Device Package	120-BGA (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/efm32gg995f1024-bga120

Bit	Name	Reset	Access	Description

Value	Mode	Description
4	WS2	Two wait-states inserted for each fetch or read transfer. This mode is required for a core frequency above 32 MHz.
5	WS2SCBTP	Two wait-state access with SCBTP enabled.

7.5.3 MSC_WRITECTRL - Write Control Register

Offset	Bit Position																													
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																											0	0	0	0	0	0
Access																											RW	RW	RW	RW	RW	RW
Name																											RWWEN	LPERASE	LPWRITE	WDOUBLE	IRQERASEABORT	WREN

Bit	Name	Reset	Access	Description
31:6	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
5	RWWEN	0	RW	**Read-While-Write Enable**
	When set, reads to the upper half of the flash can be done while writes/erases are being done in the lower half of the flash, and vice versa. Reading from the same half as a flash write/erase will stall the access until the write/erase has completed.			
4	LPERASE	0	RW	**Low-Power Erase**
	When set, the erase time doubles while halving the erase current.			
3	LPWRITE	0	RW	**Low-Power Erase**
	When set, write times might double while reducing current consumption.			
2	WDOUBLE	0	RW	**Write two words at a time**
	When set, two words are written to the flash at a time.			
1	IRQERASEABORT	0	RW	**Abort Page Erase on Interrupt**
	When this bit is set to 1, any Cortex interrupt aborts any current page erase operation.			
0	WREN	0	RW	**Enable Write/Erase Controller**
	When this bit is set, the MSC write and erase functionality is enabled.			

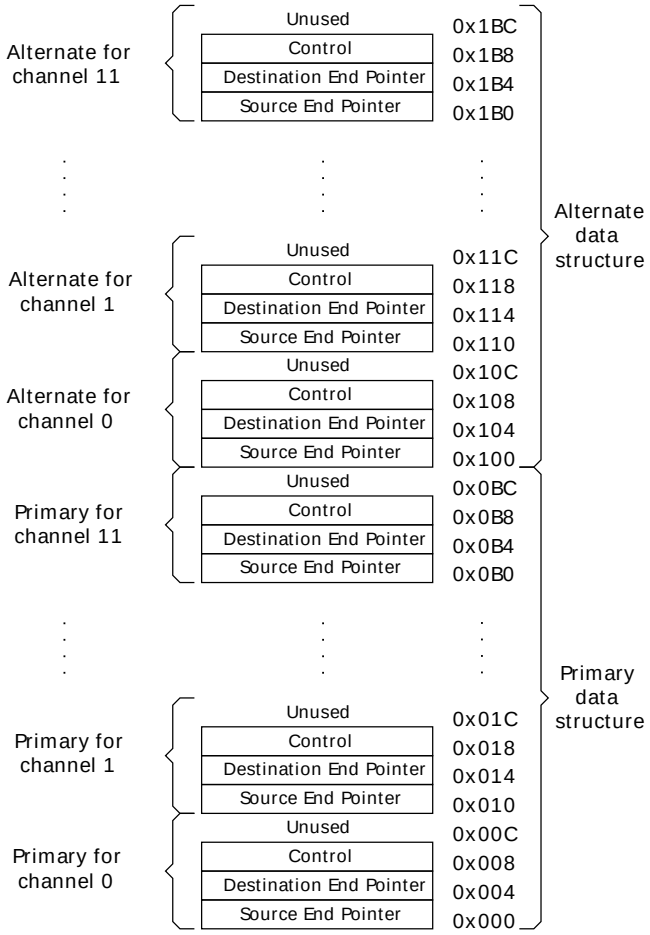
7.5.4 MSC_WRITECMD - Write Command Register

Offset	Bit Position																													
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																				0			0	0			0	0	0	0	0	0
Access																				W1			W1	W1			W1	W1	W1	W1	W1	W1
Name																				CLEARWDATA			ERASEMAIN1	ERASEMAIN0			ERASEABORT	WRITETRIG	WRITEONCE	WRITEEND	ERASEPAGE	LADDRIM

Bit	Name	Reset	Access	Description
31:13	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		

Figure 8.7 (p. 63) shows a detailed memory map of the descriptor structure.

Figure 8.7. Detailed memory map for the 12 channels, including the alternate data structure



The controller uses the system memory to enable it to access two pointers and the control information that it requires for each channel. The following subsections will describe these 32-bit memory locations and how the controller calculates the DMA transfer address.

8.4.3.1 Source data end pointer

The src_data_end_ptr memory location contains a pointer to the end address of the source data. Figure 8.7 (p. 63) lists the bit assignments for this memory location.

Table 8.7. src_data_end_ptr bit assignments

Bit	Name	Description
[31:0]	src_data_end_ptr	Pointer to the end address of the source data

Before the controller can perform a DMA transfer, you must program this memory location with the end address of the source data. The controller reads this memory location when it starts a 2^R DMA transfer.

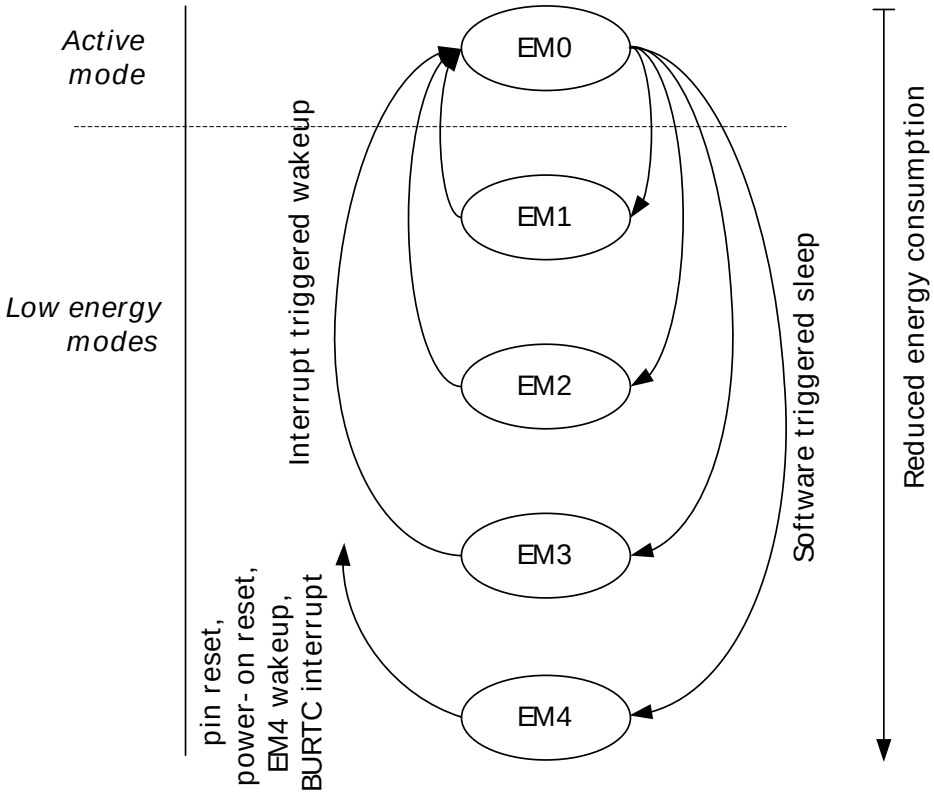
Note

The controller does not write to this memory location.

8.4.3.2 Destination data end pointer

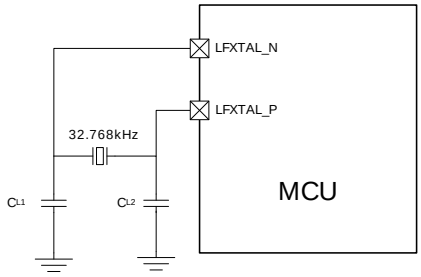
The dst_data_end_ptr memory location contains a pointer to the end address of the destination data. Table 8.8 (p. 64) lists the bit assignments for this memory location.

Figure 10.2. EMU Energy Mode Transitions



No direct transitions between EM1, EM2 or EM3 are available, as can also be seen from Figure 10.2 (p. 107) . Instead, a wakeup will transition back to EM0, in which software can enter any other low energy mode. An overview of the supported energy modes and the functionality available in each mode is shown in Table 10.1 (p. 108) . Most peripheral functionality indicated as "On" in a particular energy mode can also be turned off from software in order to save further energy.

Figure 11.5. LFXO Pin Connection



It is possible to connect an external clock source to HFXTAL_N/LFXTAL_N pin of the HFXO or LFXO oscillator. By configuring the HFXOMODE/LFXOMODE fields in CMU_CTRL, the HFXO/LFXO can be bypassed.

11.3.3.2 HFRCO, LFRCO and AUXHFRCO

The HFRCO and AUXHFRCO can be set to one of several different frequency bands from 1 MHz to 28 MHz by setting the BAND field in CMU_HFRCOCTRL and CMU_AUXHFRCOCTRL.The HFRCO and AUXHFRCO frequency bands are calibrated during production test, and the production tested calibration values can be read from the Device Information (DI) page. The DI page contains a separate tuning value for each frequency band. During reset, HFRCO and AUXHFRCO tuning values are set to the production calibrated values for the 14 MHz band, which is the default frequency band. When changing to a different HFRCO or AUXHFRCO band, make sure to also update the tuning value.

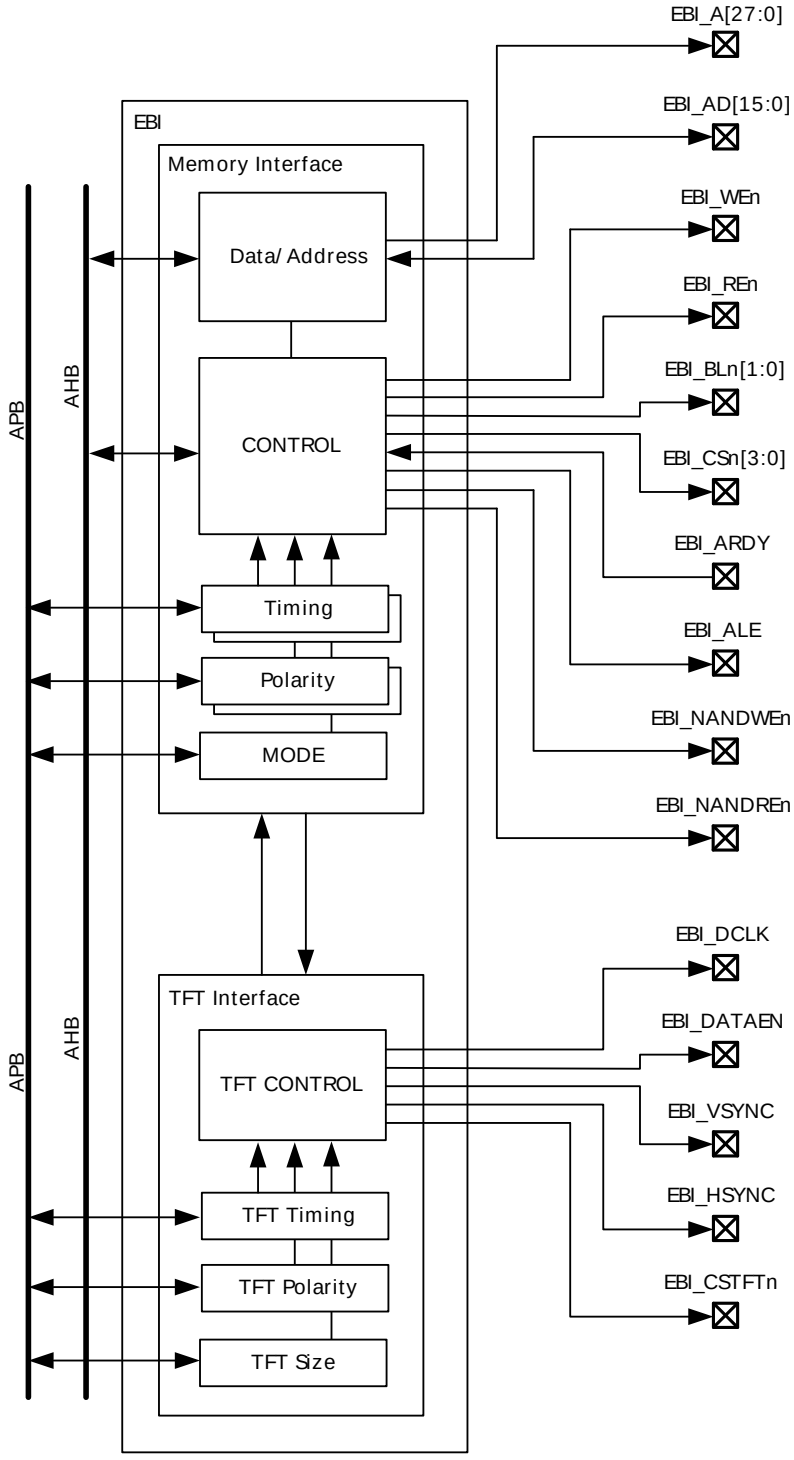
The LFRCO and is also calibrated in production and its TUNING value is set to the correct value during reset.

11.3.3.3 RC oscillator calibration

It is possible to calibrate the HFRCO, AUXHFRCO and LFRCO to achieve higher accuracy (see the device datasheets for details on accuracy). The frequency is adjusted by changing the TUNING fields in CMU_HFRCOCTRL/CMU_AUXHFRCOCTRL/CMU_LFRCOCTRL. Changing to a higher value will result in a higher frequency. Please refer to the datasheet for stepsize details.

The CMU has built-in HW support to efficiently calibrate the RC oscillators at run-time, see Figure 11.6 (p. 134) The concept is to select a reference and compare the RC frequency with the reference frequency. When the calibration circuit is started, one down-counter running on a selectable clock (DOWNSEL in CMU_CALCTRL) and one up-counter running on a selectable clock (UPSEL in CMU_CALCTRL) are started simultaneously. The top value for the down-counter must be written to CMU_CALCNT before calibration is started. The smallest value that can be written to the CMU_CALCNT is 1. The down-counter counts for CMU_CALCNT+1 cycles. When the down-counter has reached 0, the up-counter is sampled and the CALRDY interrupt flag is set. If CONT in CMU_CALCTRL is cleared, the counters are stopped at this point. If continuous mode is selected by setting CONT in CMU_CALCTRL the down-counter reloads the top value and continues counting and the up-counter restarts from 0. Software can then read out the sampled up-counter value from CMU_CALCNT. Then it is easy to find the ratio between the reference and the oscillator subject to the calibration. Overflows of the up-counter will not occur. If the up-counter reaches its top value before the down counter reaches 0, the top counter stays at its top value. Calibration can be stopped by writing CALSTOP in CMU_CMD. With this HW support, it is simple to write efficient calibration algorithms in software.

Figure 14.1. EBI Overview



14.3.1 Non-multiplexed 8-bit Data, 8-bit Address Mode

In this mode, 8-bit address and 8-bit data is supported. The address is put on the higher 8 bits of the EBI_AD lines while the data uses the lower 8 bits. This mode is set by programming the MODE field in the EBI_CTRL register to D8A8. The address space can be extended to 256 MB by using the EBI_A lines as described in Section 14.3.6 (p. 183) . Read and write signals in 8-bit mode are shown in Figure 14.2 (p. 178) and Figure 14.3 (p. 178) respectively.

respectively. For example, in case all memory banks use the 8-bit addressing mode D8A8, then the lower 8 address bits are always output on EBI_AD. Therefore, if address extension is required, only address bits 8 and upwards need to be enabled on EBI_A. This is done by setting the EBI_A lower bound to 8 by setting ALB to A8 in EBI_ROUTE and by enabling the required higher address lines via the APEN bitfield in EBI_ROUTE. The operation of the APEN and ALB bitfields is shown in Table 14.2 (p. 185) for some typical configurations.

Table 14.2. EBI Enabling EBI_ADDR lines for transaction with address Addr and data Data

Configuration	Addresses on EBI_A	Addresses/data on EBI_AD
MODE = D8A8, ALB = A8, APEN = A28	EBI_A[27:8] = Addr[27:8]	EBI_AD[15:0] = {Addr[7:0], Data[7:0]}
MODE = D16A16ALE, ALB = A16, APEN = A27	EBI_A[26:16] = Addr[27:17]	EBI_AD[15:0] = Addr[16:1]; Data[15:0]
MODE = D8A24ALE, ALB = A24, APEN = A28	EBI_A[27:24] = Addr[27:24]	EBI_AD[15:0] = Addr[23:8]; {Addr[7:0], Data[7:0]}
MODE = D16, ALB = A0, APEN = A27	EBI_A[26:0] = Addr[27:1]	EBI_AD[15:0] = Data[15:0]

14.3.7 Prefetch Unit and Write Buffer

Prefetching from external memory can enhance the performance of a sequence of consecutive transfers. In particular sequential code execution from external memory can benefit from prefetch. Also prefetch will typically lead to better utilization of intrapage accesses in case page mode is used. If prefetch is enabled, the prefetch unit will sequentially prefetch one data item of the same width as the last Cortex-M3 or DMA read transaction handled by the EBI. Note that one prefetch transaction might lead to multiple external device transactions as described in Table 14.3 (p. 188) . Prefetch is not performed in reaction to write transactions, nor will prefetch cross bank boundaries. The prefetch unit is enabled via the PREFETCH bitfield in the EBI_RDTIMING and EBI_RDTIMINGn registers. When the ITS bitfield in the EBI_CTRL register is set to 0, the PREFETCH bitfield from EBI_RDTIMING applies to all 4 memory banks. When ITS is set to 1 the prefetch unit can be individually enabled per bank. In this case register EBI_RDTIMING only applies to bank 0. Prefetch enabling for bank n is then defined in the EBI_RDTIMINGn register.

The EBI has a 1 entry 32-bit wide write buffer. The write buffer can be used to limit stalling by partially decoupling the Cortex-M3 or DMA from a potentially slow external device. Only writes which are guaranteed to not cause an error (e.g. timeout) in the EBI will be buffered when the write buffer is enabled, such that precise error generation is guaranteed. The write buffer is disabled via the WBUFDIS bitfield in the EBI_WRTIMING and EBI_WRTIMINGn registers. When the ITS bitfield in the EBI_CTRL register is set to 0, the WBUFDIS bitfield from EBI_WRTIMING applies to all 4 memory banks. When ITS is set to 1 the write buffer can be individually disabled per bank. In this case register EBI_WRTIMING only applies to bank 0. Write buffer disabling for bank n is then defined in the EBI_WRTIMINGn register.

The AHBACT status bit in the EBI_STATUS register indicates whether an AHB transaction is still active in the EBI or not. When performing an AHB write, the AHBACT bit stays 1 until the required transaction(s) with the external device have finished, independent of whether the AHB write gets buffered or not. On an AHB read with prefetching enabled, AHBACT stays high until the potential external device prefetch transaction(s) have finished.

14.3.8 Strobe length

For external devices with low, but non-zero, setup requirements the performance overhead for EBI transactions can be relatively large if a full cycle setup time needs to be used. It is possible to borrow half of the cycle time from a neighboring strobe phase in order to define setup times with a granularity of half the internal clock period.

The durations of the EBI_ALE, EBI_REn, EBI_WEn, EBI_NANDREn and EBI_NANDWEn strobes can be individually decreased by half the internal clock period via the HALFALE, HALFRE and HALFWE bitfields in the address timing, read timing and write timing registers respectively. In case of EBI_ALE

Figure 14.23. EBI Default Memory Map (ALTMAP = 0)

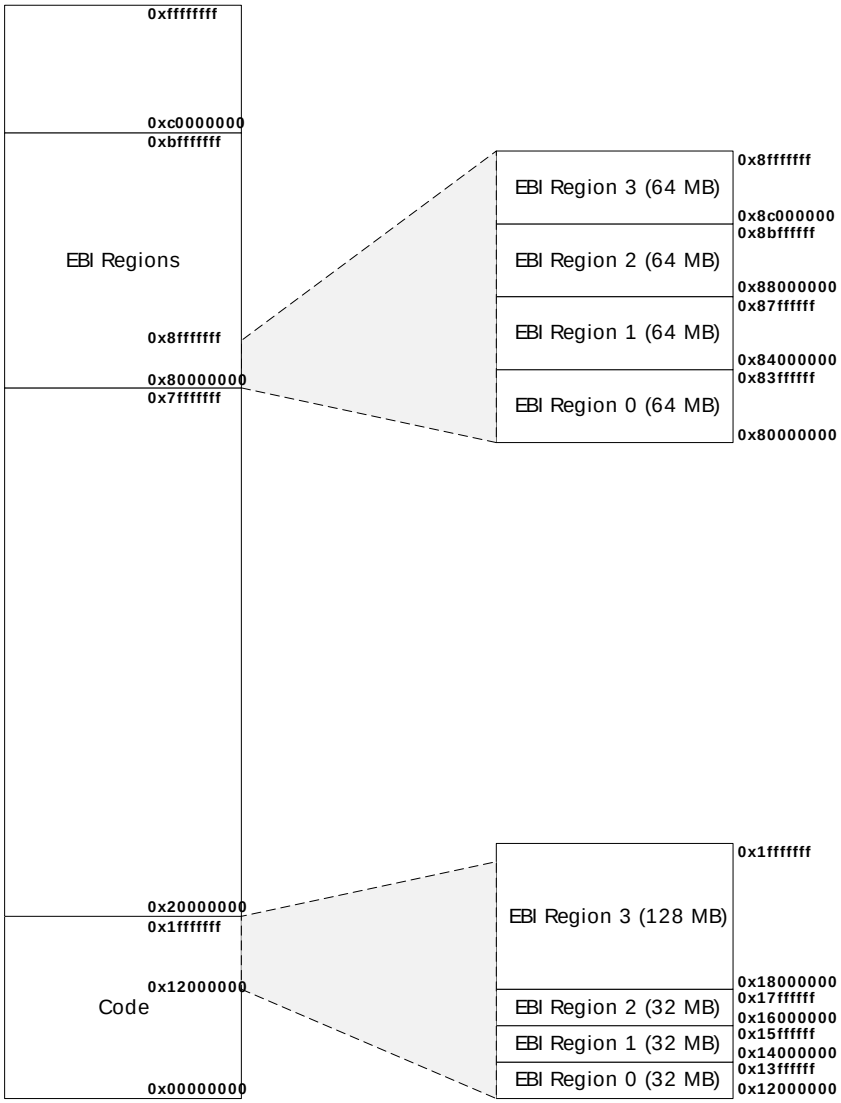


Figure 14.35. EBI TFT Direct Drive from External Memory (non-multiplexed address/data)

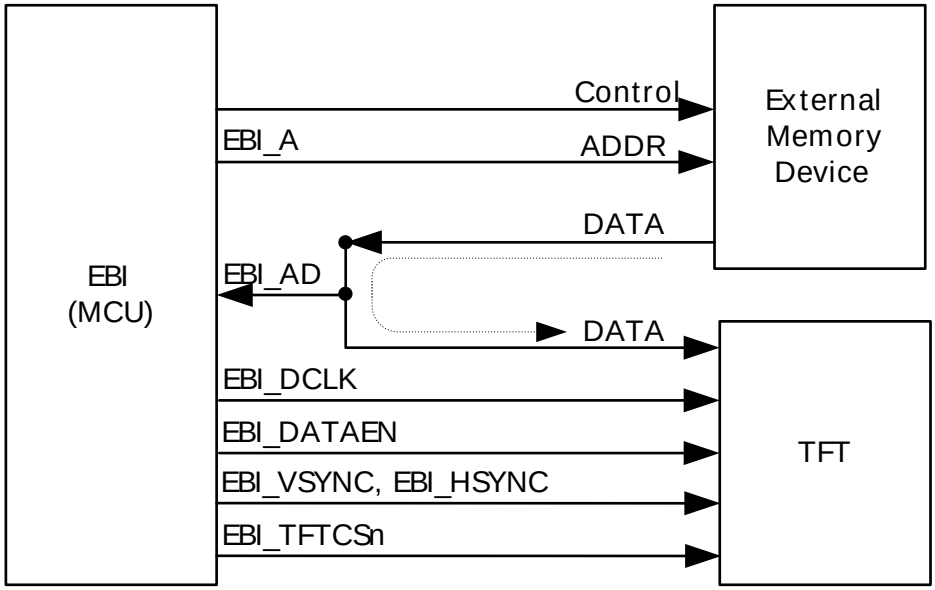
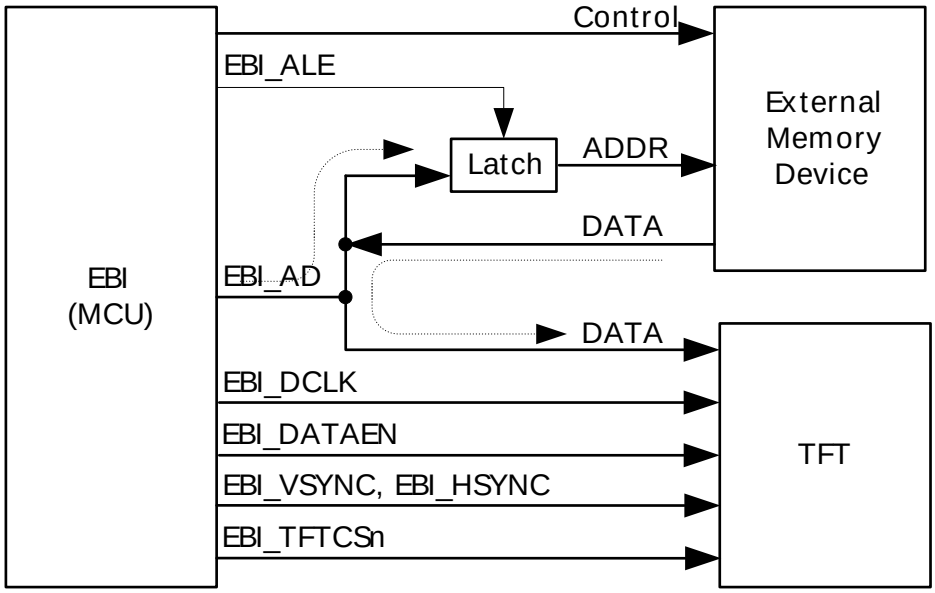


Figure 14.36. EBI TFT Direct Drive from External Memory (multiplexed address/data)



The start address for the frame transfer is defined in the EBI_TFTFRAMEBASE register. The Direct Drive address is automatically incremented for each visible pixel and it does therefore not depend on the programmed porch sizes. The address increment depends on the WIDTH bitfield in the EBI_TFTCTRL register. The increment per visible pixel is 1 if the WIDTH bitfield in the EBI_TFTCTRL register is programmed to BYTE and it is 2 if WIDTH is programmed to HALFWORD. Additionally a horizontal stride is added to the Direct Drive address at the end of each visible line. This stride can be programmed in the HSTRIDE bitfield of the EBI_TFTSTRIDE register. The first visible pixel always corresponds to the address defined in the EBI_TFTFRAMEBASE register. On either the vertical or horizontal synchronization event, as defined in the FBCTRIG bitfield of the EBI_TFTCTRL register,

6. To perform a data OUT transfer in the status OUT phase, the application must program the core as described in OUT Data Transfers in Slave and DMA Modes (p. 289) .
   - The application must program the USB_DCFG.NZSTSOUTHSHK handshake field to a proper setting before transmitting an data OUT transfer for the Status stage.
   - In DMA mode, the application must reprogram the USB_DOEPx_DMAADDR register to receive the control OUT data packet to a different memory location.
7. Assertion of the USB_DOEPx_INT.XFERCOMPL interrupt indicates completion of the status OUT phase of the control transfer. This marks the successful completion of the control read transfer.
   - To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint as explained in OUT Data Transfers in Slave and DMA Modes (p. 289) .
   - USB_DOEPx_CTL.EPENA = 1

15.4.4.2.2.3 Two-Stage Control Transfers (SETUP/Status IN)

This section describes two-stage control transfers.

Application Programming Sequence

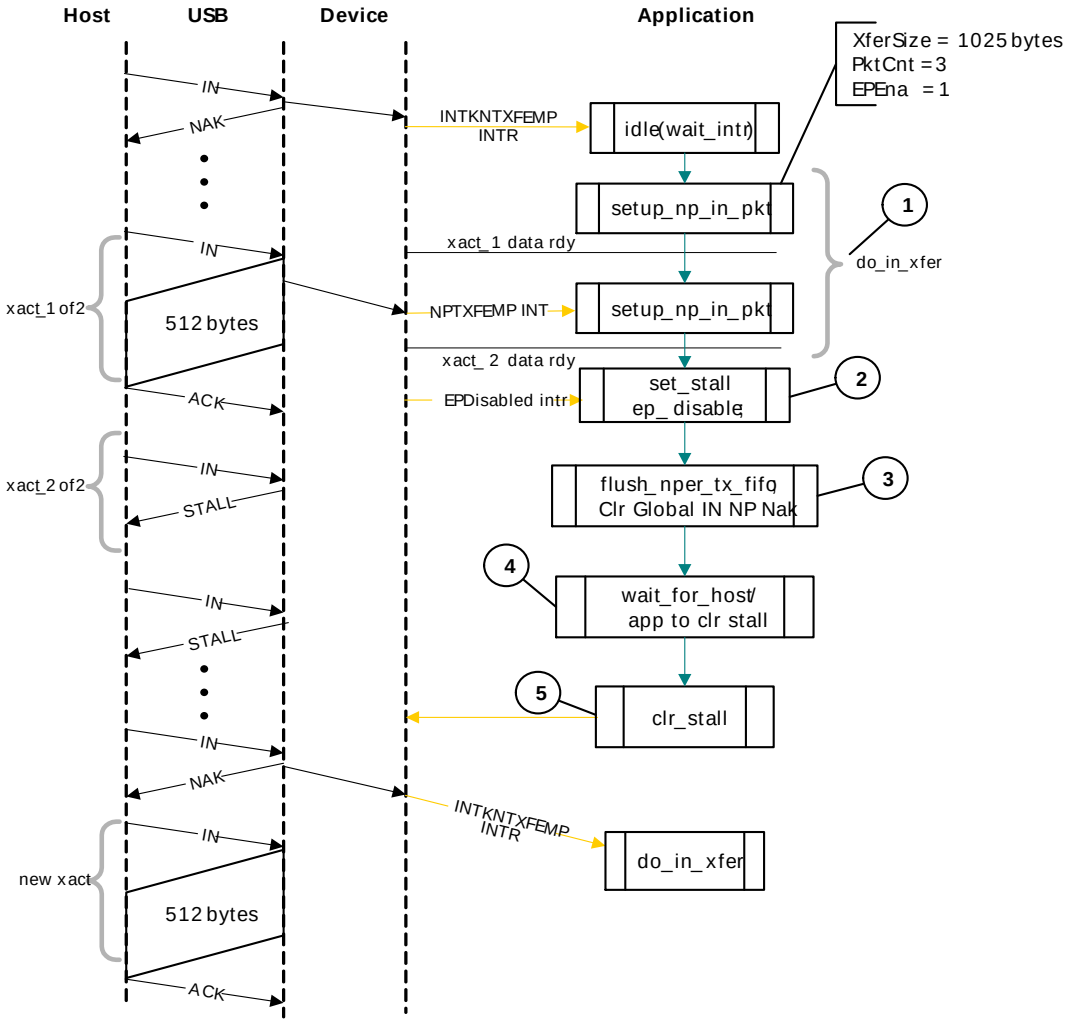
1. Assertion of the USB_DOEPx_INT.SETUP interrupt indicates that a valid SETUP packet has been transferred to the application. See OUT Data Transfers in Slave and DMA Modes (p. 289) for more detail. To receive the next SETUP packet, the application must reprogram the USB_DOEPx_TSIZ.SUPCNT field to 3 at the end of the Setup stage.
2. Decode the last SETUP packet received before the assertion of the SETUP interrupt. If the packet indicates a two-stage control command, the application must do the following.
   - To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint. See OUT Data Transfers in Slave and DMA Modes (p. 289) for details.
     - USB_DOEPx_CTL.EPENA = 1
   - Depending on the type of Setup command received, the application can be required to program registers in the core to execute the received Setup command.
3. For the status IN phase, the application must program the core described in Generic Non-Periodic (Bulk and Control) IN Data Transfers Without Thresholding in DMA and Slave Mode(p. 312) to perform a data IN transfer.
4. Assertion of the USB_DIEPx_INT.XFERCOMPL interrupt indicates the completion of the status IN phase of the control transfer.
5. The previous step must be repeated until the USB_DIEPx_INT.XFERCOMPL interrupt is detected on the endpoint, marking the completion of the two-stage control transfer.

Example: Two-Stage Control Transfer

These notes refer to Figure 15.20 (p. 294) .

1. SETUP packet #1 is received on the USB and is written to the receive FIFO, and the core responds with an ACK handshake. This handshake is lost and the host detects a timeout.
2. The SETUP packet in the receive FIFO results in a USB_GINTSTS.RXFLVL interrupt to the application, causing the application to empty the receive FIFO.
3. SETUP packet #2 on the USB is written to the receive FIFO, and the core responds with an ACK handshake.
4. The SETUP packet in the receive FIFO sends the application the USB_GINTSTS.RXFLVL interrupt and the application empties the receive FIFO.
5. After the second SETUP packet, the host sends a control IN token for the status phase. The core issues a NAK response to this token, and writes a Setup Stage Done entry to the receive FIFO. This entry results in a USB_GINTSTS.RXFLVL interrupt to the application, which empties the receive FIFO. After reading out the Setup Stage Done DWORD, the core asserts the USB_DOEPx_INT.SETUP packet interrupt to the application.
6. On this interrupt, the application processes SETUP Packet #2, decodes it to be a two-stage control command, and clears the control IN NAK bit.

Figure 15.26. USBTRDTIM Max Timing Case ERROR wrong image



15.4.4.2.3.10 Handling Babble Conditions

If receives a packet that is larger than the maximum packet size for that endpoint, the core stops writing data to the Rx buffer and waits for the end of packet (EOP). When the core detects the EOP, it flushes the packet in the Rx buffer and does not send any response to the host.

If the core continues to receive data at the EOF2 (the end of frame 2, which is very close to SOF), the core generates an early_suspend interrupt (USB_GINTSTS.ERLYSUSP). On receiving this interrupt, the application must check the erratic_error status bit (USB_DSTS.ERRTICERR). If this bit is set, the application must take it as a long babble and perform a soft reset.

15.4.4.2.3.11 Generic Non-Periodic (Bulk and Control) IN Data Transfers in DMA and Slave Mode

To initialize the core after power-on reset, the application must follow the sequence in Overview: Programming the Core (p. 250) . Before it can communicate with the host, it must initialize an endpoint as described in Endpoint Initialization (p. 285) . For packet writes in Slave mode, see: Packet Write in Slave Mode (p. 307) .

Application Requirements

1. Before setting up an IN transfer, the application must ensure that all data to be transmitted as part of the IN transfer is part of a single buffer, and must program the size of that buffer and its start address (in DMA mode) to the endpoint-specific registers.
2. For IN transfers, the Transfer Size field in the Endpoint Transfer Size register denotes a payload that constitutes multiple maximum-packet-size packets and a single short packet. This short packet is transmitted at the end of the transfer.

When enabling the I^2C, the ABORT command or the Bus Idle Timeout feature must be applied prior to use even if the BUSY flag is not set.

16.3.3 Safely Disabling and Changing Slave Configuration

The I^2C slave is partially asynchronous, and some precautions are necessary to always ensure a safe slave disable or slave configuration change. These measures should be taken, if (while the slave is enabled) the user cannot guarantee that an address match will not occur at the exact time of slave disable or slave configuration change.

Worst case consequences for an address match while disabling slave or changing configuration is that the slave may end up in an undefined state. To reset the slave back to a known state, the EN bit in I2Cn_CTRL must be reset. This should be done regardless of whether the slave is going to be re-enabled or not.

16.3.4 Clock Generation

The SCL signal generated by the I^2C master determines the maximum transmission rate on the bus. The clock is generated as a division of the peripheral clock, and is given by Equation 16.2 (p. 420) :

I^2C Maximum Transmission Rate

$$f_{SCL} = 1/(T_{low} + T_{high}), \quad (16.2)$$

where

T_{low} and T_{high} is the low and high periods of the clock signal respectively, given below. When the clock is not streched, the low and high periods of the clock signal are:

I^2C High and Low Cycles Equations

$$T_{high} = (N_{high} \times (CLKDIV + 1))/f_{HFPERCLK},$$
$$T_{low} = (N_{low} \times (CLKDIV + 1))/f_{HFPERCLK}. \quad (16.3)$$

Equation 16.3 (p. 420) and Equation 16.2 (p. 420) does not apply for low clock division factors (0, 1 and 2) because of synchronization. For these clock division factors, the formulas for computing high and low periods of the clock signal are given in Table 16.2 (p. 420) .

Table 16.2. I^2C High and Low Periods for Low CLKDIV

CLKDIV	Standard (4:4)		Asymmetric (6:3)		Fast (11:6)	
	T_{low}	T_{high}	T_{low}	T_{high}	T_{low}	T_{high}
0	$7/f_{HFPERCLK}$	$7/f_{HFPERCLK}$	$9/f_{HFPERCLK}$	$6/f_{HFPERCLK}$	$14/f_{HFPERCLK}$	$9/f_{HFPERCLK}$
1	$10/f_{HFPERCLK}$	$10/f_{HFPERCLK}$	$14/f_{HFPERCLK}$	$8/f_{HFPERCLK}$	$24/f_{HFPERCLK}$	$14/f_{HFPERCLK}$
2	$15/f_{HFPERCLK}$	$15/f_{HFPERCLK}$	$21/f_{HFPERCLK}$	$12/f_{HFPERCLK}$	$36/f_{HFPERCLK}$	$21/f_{HFPERCLK}$

The values of N_{low} and N_{high} and thus the ratio between the high and low parts of the clock signal is controlled by CLHR in the I2Cn_CTRL register. The available modes are summarized in Table 16.3 (p. 421) along with the highest I^2C-bus frequencies in the given modes that can be achieved without violating the timing specifications of the I^2C-bus. The maximum data hold time is dependent on the DIV and is given by:

Maximum Data Hold Time

$$t_{HD,DAT\text{-}max} = (4+DIV)/f_{HFPERCLK}. \quad (16.4)$$

Note

DIV must be set to 1 or higher during slave mode operation.

17.4 Register Map

The offset register address is relative to the registers base address.

Offset	Name	Type	Description
0x000	USARTn_CTRL	RW	Control Register
0x004	USARTn_FRAME	RW	USART Frame Format Register
0x008	USARTn_TRIGCTRL	RW	USART Trigger Control register
0x00C	USARTn_CMD	W1	Command Register
0x010	USARTn_STATUS	R	USART Status Register
0x014	USARTn_CLKDIV	RW	Clock Control Register
0x018	USARTn_RXDATAX	R	RX Buffer Data Extended Register
0x01C	USARTn_RXDATA	R	RX Buffer Data Register
0x020	USARTn_RXDOUBLEX	R	RX Buffer Double Data Extended Register
0x024	USARTn_RXDOUBLE	R	RX FIFO Double Data Register
0x028	USARTn_RXDATAXP	R	RX Buffer Data Extended Peek Register
0x02C	USARTn_RXDOUBLEXP	R	RX Buffer Double Data Extended Peek Register
0x030	USARTn_TXDATAX	W	TX Buffer Data Extended Register
0x034	USARTn_TXDATA	W	TX Buffer Data Register
0x038	USARTn_TXDOUBLEX	W	TX Buffer Double Data Extended Register
0x03C	USARTn_TXDOUBLE	W	TX Buffer Double Data Register
0x040	USARTn_IF	R	Interrupt Flag Register
0x044	USARTn_IFS	W1	Interrupt Flag Set Register
0x048	USARTn_IFC	W1	Interrupt Flag Clear Register
0x04C	USARTn_IEN	RW	Interrupt Enable Register
0x050	USARTn_IRCTRL	RW	IrDA Control Register
0x054	USARTn_ROUTE	RW	I/O Routing Register
0x058	USARTn_INPUT	RW	USART Input Register
0x05C	USARTn_I2SCTRL	RW	I2S Control Register

17.5 Register Description

17.5.1 USARTn_CTRL - Control Register

Offset	Bit Position																														
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset		0	0	0	0x0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0x0		0	0	0	0	0
Access		RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW
Name		MVDIS	AUTOTX	BYTESWAP	TXDELAY			ERRSTX	ERRSRX	ERRSDMA	BIT8DV	SKIPPERRF	SCRETRANS	SCMODE	AUTOTRI	AUTOCS	CSINV	TXINV	RXINV	TXBIL	CSMA	MSBF	CLKPHA	CLKPOL		OVS		MPAB	MPM	CCEN	LOOPBK	SYNC

Bit	Name	Reset	Access	Description
31	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
30	MVDIS	0	RW	**Majority Vote Disable**
	Disable majority vote for 16x, 8x and 6x oversampling modes.			
29	AUTOTX	0	RW	**Always Transmit When RX Not Full**

Bit	Name	Reset	Access	Description
	Set when the value written to TXDATA is being synchronized.			
5	TXDATAX	0	R	**TXDATAX Register Busy**
	Set when the value written to TXDATAX is being synchronized.			
4	SIGFRAME	0	R	**SIGFRAME Register Busy**
	Set when the value written to SIGFRAME is being synchronized.			
3	STARTFRAME	0	R	**STARTFRAME Register Busy**
	Set when the value written to STARTFRAME is being synchronized.			
2	CLKDIV	0	R	**CLKDIV Register Busy**
	Set when the value written to CLKDIV is being synchronized.			
1	CMD	0	R	**CMD Register Busy**
	Set when the value written to CMD is being synchronized.			
0	CTRL	0	R	**CTRL Register Busy**
	Set when the value written to CTRL is being synchronized.			

19.5.19 LEUARTn_ROUTE - I/O Routing Register

Offset	Bit Position																											
0x054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																							0x0								0	0
Access																							RW								RW	RW
Name																							LOCATION								TXPEN	RXPEN

Bit	Name	Reset	Access	Description
31:11	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
10:8	LOCATION	0x0	RW	**I/O Location**

Decides the location of the LEUART I/O pins.

Value	Mode	Description
0	LOC0	Location 0
1	LOC1	Location 1
2	LOC2	Location 2
3	LOC3	Location 3
4	LOC4	Location 4

7:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	TXPEN	0	RW	**TX Pin Enable**

When set, the TX pin of the LEUART is enabled.

Value	Description
0	The LEUn_TX pin is disabled
1	The LEUn_TX pin is enabled

0	RXPEN	0	RW	**RX Pin Enable**

When set, the RX pin of the LEUART is enabled.

Value	Description
0	The LEUn_RX pin is disabled
1	The LEUn_RX pin is enabled

The direction of the quadrature code and control of the counter is generated by the simple binary function outlined by Table 24.1 (p. 611) . Note that this function also filters some invalid inputs that may occur when the shaft changes direction or temporarily toggles direction.

Table 24.1. PCNT QUAD Mode Counter Control Function

Inputs		Control/Status	
S1IN posedge	S1IN negedge	Count Enable	CNTDIR status bit
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	0

Note

PCNTn_S1IN is sampled on both edges of PCNTn_S0IN.

24.3.2 Hysteresis

By default the pulse counter wraps to 0 when passing the configured top value, and wraps to the top value when counting down from 0. On these events, a system will likely want to wake up to store and track the overflow count. This is fine if the pulse counter is tracking a monotonic value or a value that does not change directions frequently. If you have the latter however, and the counter changes directions around the overflow/underflow point, the system will have to wake up a lot to keep track of the rotations, causing high current consumptions

To solve this, the pulse counter has a way of introducing hysteresis to the counter. When HYST in PCNTn_CTRL is set, the pulse counter will always wrap to TOP/2 on underflows and overflows. This takes the counter away from the area where it might overflow or underflow, removing the problem.

Given a starting value of 0 for the counter, the absolute count value when hysteresis is enabled can be calculated with the equations Equation 24.1 (p. 611) or Equation 24.2 (p. 611) , depending on whether the TOP value is even or odd.

Absolute position with hysteresis and even TOP value

$$CNT_{abs} = CNT - UF_{CNT} \times (TOP/2+1) + OF_{CNT} \times (TOP/2+1) \quad (24.1)$$

Absolute position with hysteresis and odd TOP value

$$CNT_{abs} = CNT - UF_{CNT} \times (TOP/2+1) + OF_{CNT} \times (TOP/2+2) \quad (24.2)$$

24.3.3 Auxiliary counter

To be able to keep explicit track of counting in one direction in addition to the regular counter which counts both up and down, the auxiliary counter can be used. The pulse counter can for instance be configured to keep track of the absolute rotation of the wheel, and at the same time the auxiliary counter can keep track of how much the wheel has reversed.

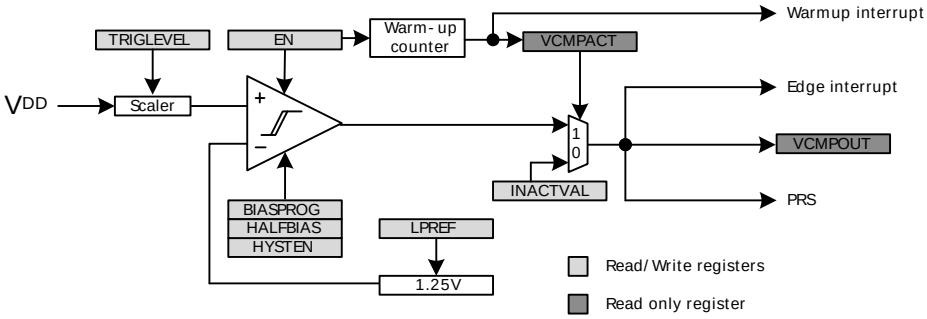
The auxiliary counter is enabled by configuring AUXCNTEV in PCNTn_CTRL. It will always count up, but it can be configured whether it should count up on up-events, down-events or both, keeping track of rotation either way or general movement. The value of the auxiliary counter can be read from the PCNTn_AUXCNT register.

Overflows on the auxiliary counter happen when the auxiliary counter passes the top value of the pulse counter, configured in PCNTn_TOP. In that event, the AUXOF interrupt flag is set, and the auxiliary counter wraps to 0.

27.3 Functional Description

An overview of the VCMP is shown in Figure 27.1 (p. 681) .

Figure 27.1. VCMP Overview



The comparator has two analog inputs, one positive and one negative. When the comparator is active, the output indicates which of the two input voltages is higher. When the voltage on the positive input is higher than the negative input voltage, the digital output is high and vice versa.

The output of the comparator can be read in the VCMPOUT bit in VCMP_STATUS. Configuration registers should only be changed while the comparator is disabled.

27.3.1 Warm-up Time

VCMP is enabled by setting the EN bit in VCMP_CTRL. When this bit is set, the comparator must stabilize before becoming active and the outputs can be used. This time period is called the warm-up time. The warm-up time is a configurable number of HFPERCLK cycles, set in WARMTIME, which should be set to at least 10 µs. When the comparator is enabled and warmed up, the VCMPACT bit in VCMP_STATUS will be set to indicate that the comparator is active.

As long as the comparator is not enabled or not warmed up, VCMPACT will be cleared and the comparator output value is set to the value in INACTVAL in VCMP_CTRL.

One should wait until the warm-up period is over before entering EM2 or EM3, otherwise no comparator interrupts will be detected. EM1 can still be entered during warm-up. After the warm-up period is completed, interrupts will be detected in EM2 and EM3.

27.3.2 Response Time

There is a delay from when the actual input voltage changes polarity, to when the output toggles. This period is called the response time and can be altered by increasing or decreasing the bias current to the comparator through the BIAS and HALFBIAS fields in VCMP_CTRL as shown in Table 27.1 (p. 681) . Setting a lower bias current will result in lower power consumption, but a longer response time.

Table 27.1. Bias Configuration

BIAS	Bias Current (µA)	
	HALFBIAS=0	HALFBIAS=1
0b0000	0.1	0.05
0b0001	0.2	0.1
0b0010	0.4	0.2
0b0011	0.6	0.3

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	VCMPOUT	0	R	**Voltage Supply Comparator Output**
	Voltage supply comparator output value			
0	VCMPACT	0	R	**Voltage Supply Comparator Active**
	Voltage supply comparator active status.			

27.5.4 VCMP_IEN - Interrupt Enable Register

Offset	Bit Position																													
0x00C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															RW	RW
Name																															WARMUP	EDGE

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	WARMUP	0	RW	**Warm-up Interrupt Enable**
	Enable/disable interrupt on finished warm-up.			
0	EDGE	0	RW	**Edge Trigger Interrupt Enable**
	Enable/disable edge triggered interrupt.			

27.5.5 VCMP_IF - Interrupt Flag Register

Offset	Bit Position																													
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															R	R
Name																															WARMUP	EDGE

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	WARMUP	0	R	**Warm-up Interrupt Flag**
	Indicates that warm-up has finished.			
0	EDGE	0	R	**Edge Triggered Interrupt Flag**
	Indicates that there has been a rising and/or falling edge on the VCMP output.			

Bit	Name	Reset	Access	Description

Value	Mode	Description
9	X1024	1024 samples for each conversion result
10	X2048	2048 samples for each conversion result
11	X4096	4096 samples for each conversion result

23:21 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

20:16 TIMEBASE 0x1F RW **Time Base**

Set time base used for ADC warm up sequence according to the HFPERCLK frequency. The time base is defined as a number of HFPERCLK cycles which should be set equal to or higher than 1us.

Value	Description
TIMEBASE	ADC warm-up is set to TIMEBASE+1 HFPERCLK clock cycles and bandgap warm-up is set to 5x(TIMEBASE+1) HFPERCLK cycles.

15 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

14:8 PRESC 0x00 RW **Prescaler Setting**

Select clock division factor.

Value	Description
PRESC	Clock division factor of PRESC+1.

7:6 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

5:4 LPFMODE 0x0 RW **Low Pass Filter Mode**

These bits control the filtering of the ADC input. Details on the filter characteristics can be found in the device datasheets.

Value	Mode	Description
0	BYPASS	No filter or decoupling capacitor
1	DECAP	On chip decoupling capacitor selected
2	RCFILT	On chip RC filter selected

3 TAILGATE 0 RW **Conversion Tailgating**

Enable/disable conversion tailgating.

Value	Description
0	Scan sequence has priority, but can be delayed by ongoing single samples.
1	Scan sequence has priority and single samples will only start immediately after scan sequence.

2 *Reserved* *To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*

1:0 WARMUPMODE 0x0 RW **Warm-up Mode**

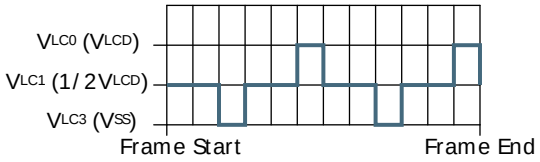
Select Warm-up Mode for ADC

Value	Mode	Description
0	NORMAL	ADC is shut down after each conversion
1	FASTBG	Bandgap references do not need warm up, but have reduced accuracy.
2	KEEPSCANREFWARM	Reference selected for scan mode is kept warm.
3	KEEPADCWARM	ADC is kept warmed up and scan reference is kept warm

28.5.2 ADCn_CMD - Command Register

Offset	Bit Position																															
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																													0	0	0	0
Access																													W1	W1	W1	W1
Name																													SCANSTOP	SCANSTART	SINGLESTOP	SINGLESTART

Figure 33.19. LCD 1/2 Bias and Triplex Multiplexing - LCD_COM2



1/2 bias and triplex multiplexing - LCD_SEG0

The LCD_SEG0 waveform on the left is just an example to illustrate how different segment waveforms can be multiplexed with the COM lines in order to turn on and off LCD pixels. As illustrated in the figures below, this waveform will turn ON pixels connected to LCD_COM1, while pixels connected to LCD_COM0 and LCD_COM2 will be turned OFF.

Figure 33.20. LCD 1/2 Bias and Triplex Multiplexing - LCD_SEG0

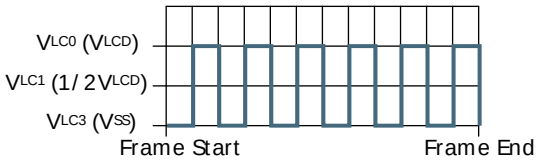
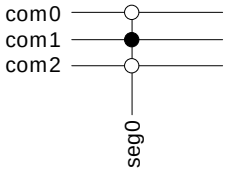


Figure 33.21. LCD 1/2 Bias and Triplex Multiplexing - LCD_SEG0 Connection



1/2 bias and triplex multiplexing - LCD_SEG0-LCD_COM0

- DC voltage = 0 (over one frame)
- $V_{RMS} = 0.4 \times V_{LCD\_OUT}$
- The LCD display pixel that is connected to LCD_SEG0 and LCD_COM0 will be OFF with this waveform

Figure 33.22. LCD 1/2 Bias and Triplex Multiplexing - LCD_SEG0-LCD_COM0

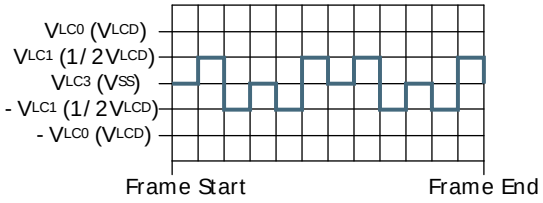


Figure 33.37. LCD 1/3 Bias and Quadruplex Multiplexing - LCD_SEG0

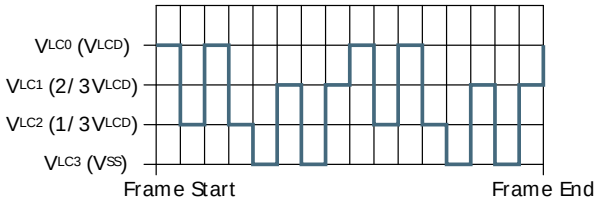
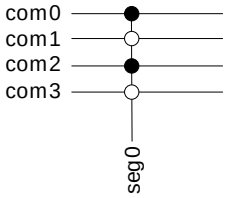


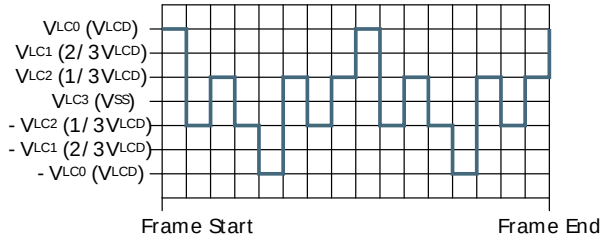
Figure 33.38. LCD 1/3 Bias and Quadruplex Multiplexing - LCD_SEG0 Connection



1/3 bias and quadruplex multiplexing - LCD_SEG0-LCD_COM0

- DC voltage = 0 (over one frame)
- $V_{RMS} = 0.58 \times V_{LCD\_OUT}$
- The LCD display pixel that is connected to LCD_SEG0 and LCD_COM0 will be ON with this waveform

Figure 33.39. LCD 1/3 Bias and Quadruplex Multiplexing - LCD_SEG0-LCD_COM0



1/3 bias and quadruplex multiplexing - LCD_SEG0-LCD_COM1

- DC voltage = 0 (over one frame)
- $V_{RMS} = 0.33 \times V_{LCD\_OUT}$
- The LCD display pixel that is connected to LCD_SEG0 and LCD_COM1 will be OFF with this waveform

34 Revision History

34.1 Revision 1.20

April 28th, 2016

Updated memory system map

Added revision E

Replaced static bit write instruction with reference to the Cortex-M3 manual

Updated GPIO pin configuration schematic

Corrected UD, LB and DI flash addresses in the MSC section.

Added full wafer as package option.

Corrected bit alignment in PID0 register in section 3.

Changes in the I^2C section

- Updated note.

- Updated Clock Generation section.

Corrected typos and added notes in the DMA Controller section.

Updated EMU Backup power domain section.

Updated the register description of LEUARTn_CTRL.

Corrected the DAC f_{sine} equation.

Added and modified notes in the WDOG Clock Source and Register Access sections.

Modified a note in the PCNT Clock Sources section.

Updated the register description of MSC_WDATA.

Updated the register description of LESENSE_BIASCTRL.

Updated the register description of BURTC_CTRL.

Updated the register descriptions of USARTn_IF, USARTn_TXDATAX and USARTn_TXDOUBLEX.

Updated the register descriptions of CMU_CTRL and CMU_CMD.

Updated the Block Diagram.

Updated the MSC Erase and Write Operations section.

34.2 Revision 1.10

July 2nd, 2014

Updated current numbers and voltage supply range.

Updated block diagram.

Moved chapter "Device Revision" to section 3.