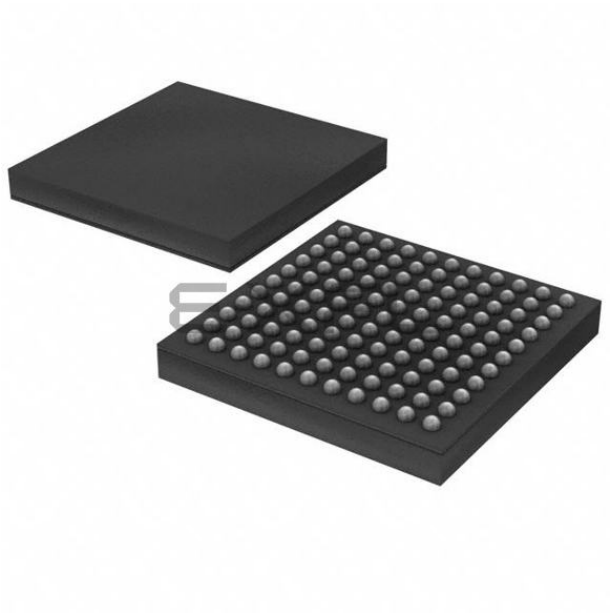
Silicon Labs - EFM32GG995F1024G-E-BGA120R Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

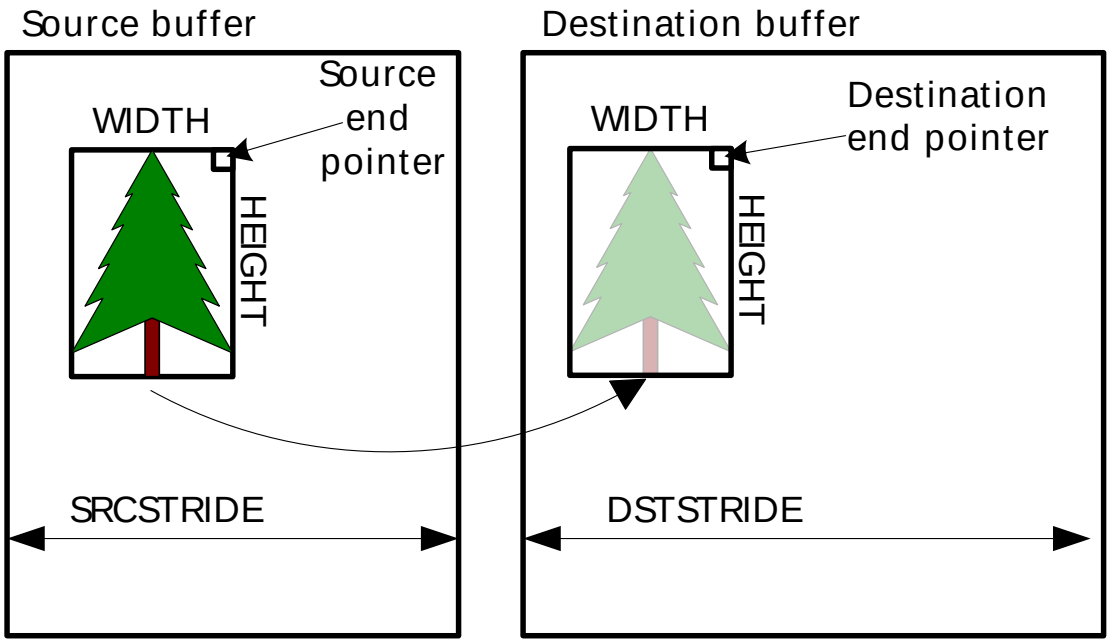
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	Active
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)	1.98V ~ 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/efm32gg995f1024g-e-bga120r

set SRCSTRIDE in DMA_RECT0 to the outer rectangle width of the source, and DSTSTRIDE in DMA_RECT0 to the outer rectangle width of the destination rectangle. Finally, the channel descriptor for channel 0 has to be configured. The source and destination end pointers should be set to the last element of the first line of the source data and destination data respectively. The number of elements to be transferred, n_minus_1 should be set equal to WIDTH in DMA_LOOP0. The parameters are visualized in Figure 8.9 (p. 69) .

Figure 8.9. 2D copy



When doing a rectangle copy, the source and destination address of the channel descriptor will be incremented line for line as the DMA works its way through the rectangle. The operation is done when the number of lines specified by HEIGHT in DMA_RECT0 has been copied. The source and destination addresses in the channel descriptor will then point at the last element of the source and destination rectangles.

On completion, the DONE interrupt flag of channel 0 is set. Looping is not supported for rectangle copy.

In some cases, e.g. when performing graphics operations, it is desirable to create a list of copy operations and have them executed automatically. This can be done using 2D copy together with the scatter gather mode of the DMA controller. Set DESCRECT in DMA_CTRL to override SCRSTRIDE and HEIGHT in DMA_RECT0 and WIDTH in DMA_LOOP0 by the values in the user part of the DMA descriptor as shown in Table 8.12 (p. 69). In this way every copy command in the list can specify these parameters individually.

Table 8.12. User data assignments when DESCRECT is set

Bit	Field	Description
[30:20]	SRCSTRIDE	Stride in source buffer
[19:10]	HEIGHT	Height - 1 of data to be copied
[9:0]	WIDTH	Width - 1 of data to be copied

With regular 2D copy, the DMA descriptor will be updated as the copy operation proceeds. To be able to reuse the 2D copy scatter gather list without rewriting source and destination end addresses, set PRDU

Bit	Name	Reset	Access	Description

Value	Mode	Description
1	ACTIVEHIGH	HSYNC is active high.

Bit	Name	Reset	Access	Description
2	DATAENPOL	0	RW	**TFT DATAEN Polarity**

Sets the polarity of the EBI_DATAEN line.

Value	Mode	Description
0	ACTIVELOW	DATAEN is active low.
1	ACTIVEHIGH	DATAEN is active high.

Bit	Name	Reset	Access	Description
1	DCLKPOL	0	RW	**TFT DCLK Polarity**

Sets the active edge polarity of the EBI_DCLK line.

Value	Mode	Description
0	ACTIVEFALLING	DCLK falling edge is the active edge.
1	ACTIVERISING	DCLK rising edge the active edge.

Bit	Name	Reset	Access	Description
0	CSPOL	0	RW	**TFT Chip Select Polarity**

Sets the polarity of the EBI_CSTFT line.

Value	Mode	Description
0	ACTIVELOW	CSTFT is active low.
1	ACTIVEHIGH	CSTFT is active high.

14.5.33 EBI_TFTDD - TFT Direct Drive Data Register

Offset	Bit Position																															
0x080	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																	0x0000															
Access																	RW															
Name																	DATA															

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15:0	DATA	0x0000	RW	**TFT Direct Drive Data from Internal Memory**

Sets the RGB value used when Direct Drive from internal memory is used (DD = INTERNAL)

14.5.34 EBI_TFTALPHA - TFT Alpha Blending Register

Offset	Bit Position																															
0x084	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																								0x000								
Access																								RW								
Name																								ALPHA								

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	DDJIT	0	R	**Direct Drive Jitter Interrupt Flag**
	Set when DCLKPERIOD is not met.			
4	DDEMPTY	0	R	**Direct Drive Data Empty Interrupt Flag**
	Set when Direct Drive engine EBI_TFTDD data is empty.			
3	VFPORCH	0	R	**Vertical Front Porch Interrupt Flag**
	Set at beginning of Vertical Front Porch.			
2	VBPORCH	0	R	**Vertical Back Porch Interrupt Flag**
	Set at end of Vertical Back Porch.			
1	HSYNC	0	R	**Horizontal Sync Interrupt Flag**
	Set at Horizontal Sync pulse.			
0	VSYNC	0	R	**Vertical Sync Interrupt Flag**
	Set at Vertical Sync pulse.			

14.5.40 EBI_IFS - Interrupt Flag Set Register

Offset	Bit Position																													
0x09C	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																											W1	W1	W1	W1	W1	W1
Name																											DDJIT	DDEMPTY	VFPORCH	VBPORCH	HSYNC	VSYNC

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	DDJIT	0	W1	**Direct Drive Jitter Interrupt Flag Set**
	Write to 1 to set Direct Drive Jitter Interrupt flag.			
4	DDEMPTY	0	W1	**Direct Drive Data Empty Interrupt Flag Set**
	Write to 1 to set Direct Drive Data Empty Interrupt flag.			
3	VFPORCH	0	W1	**Vertical Front Porch Interrupt Flag Set**
	Write to 1 to set Vertical Front Porch Interrupt flag.			
2	VBPORCH	0	W1	**Vertical Back Porch Interrupt Flag Set**
	Write to 1 to set Vertical Back Porch Interrupt flag.			
1	HSYNC	0	W1	**Horizontal Sync Interrupt Flag Set**
	Write to 1 to set Horizontal Sync interrupt flag.			
0	VSYNC	0	W1	**Vertical Sync Interrupt Flag Set**
	Write to 1 to set Vertical Sync interrupt flag.			

1. Initialize channel 2 as explained in Channel Initialization (p. 256) .
2. Set the USB_HC2_CHAR.CHENA bit to write an IN request to the Non-periodic Request Queue.
3. The core attempts to send an IN token after completing the current OUT transaction.
4. The core generates an RXFLVL interrupt as soon as the received packet is written to the receive FIFO.
5. In response to the RXFLVL interrupt, mask the RXFLVL interrupt and read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. Following this, unmask the RXFLVL interrupt.
6. The core generates the RXFLVL interrupt for the transfer completion status entry in the receive FIFO.
7. The application must read and ignore the receive packet status when the receive packet status is not an IN data packet (USB_GRXSTSR.PKTSTS != 0b0010).
8. The core generates the XFERCOMPL interrupt as soon as the receive packet status is read.
9. In response to the XFERCOMPL interrupt, disable the channel (see Halting a Channel (p. 257)) and stop writing the USB_HC2_CHAR register for further requests. The core writes a channel disable request to the non-periodic request queue as soon as the USB_HC2_CHAR register is written.
10. The core generates the RXFLVL interrupt as soon as the halt status is written to the receive FIFO.
11. Read and ignore the receive packet status.
12. The core generates a CHHLTD interrupt as soon as the halt status is popped from the receive FIFO.
13. In response to the CHHLTD interrupt, de-allocate the channel for other transfers.

Note

For Bulk/Control IN transfers, the application must write the requests when the Request queue space is available, and until the XFERCOMPL interrupt is received.

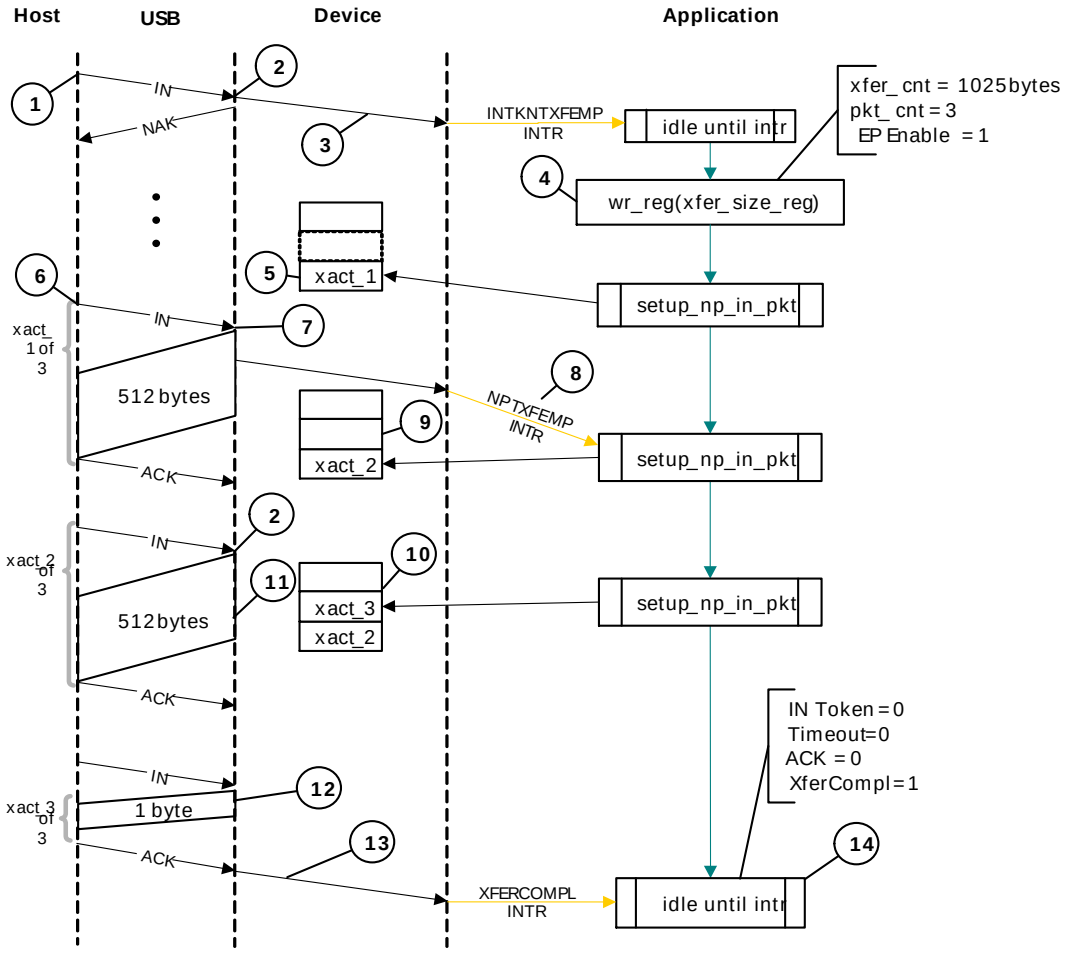
15.4.3.6.5.2 Handling Interrupts

The channel-specific interrupt service routine for bulk and control IN transactions in Slave mode is shown in the following code samples.

Interrupt Service Routine for Bulk/Control IN Transactions in Slave Mode

```
Unmask (XACTERR/XFERCOMPL/BBLERR/STALL/DATATGLERR)
if (XFERCOMPL)
{
    Reset Error Count
    Unmask CHHLTD
    Disable Channel
    Reset Error Count
    Mask ACK
}
else if (XACTERR or BBLERR or STALL)
{
    Unmask CHHLTD
    Disable Channel
    if (XACTERR)
    {
        Increment Error Count
        Unmask ACK
    }
}
else if (CHHLTD)
{
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
    {
        De-allocate Channel
    }
    else
    {
        Re-initialize Channel
    }
```

Figure 15.28. Slave Mode Bulk IN Transfer (Pipelined Transaction)



Slave Mode Bulk IN Two-Endpoint Transfer

These notes refer to Figure 15.29 (p. 317)

1. The host attempts to read data (IN token) from endpoint 1.
2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 1, and generates a USB_DIEP1_INT.INTKNTXFEMP (In Token Received When TxFIFO Empty) interrupt.
3. The application processes the interrupt and initializes USB_DIEP1_TSIZ register with the Transfer Size and Packet Count fields. The application starts writing the transaction data to the transmit FIFO.
4. The application writes one maximum packet size or less of data for endpoint 1 to the Non-periodic TxFIFO.
5. Meanwhile, the host attempts to read data (IN token) from endpoint 2.
6. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 2, and the core generates a USB_DIEP2_INT.INTKNTXFEMP (In Token Received When TxFIFO Empty) interrupt.
7. Because the application has completed writing the packet for endpoint 1, it initializes the USB_DIEP2_TSIZ register with the Transfer Size and Packet Count fields. The application starts writing the transaction data into the transmit FIFO for endpoint 2.
8. The host repeats its attempt to read data (IN token) from endpoint 1.
9. Because data is now ready in the TxFIFO, the core returns the data, which the host ACKs.
10. Meanwhile, the application has initialized the data for the next two packets in the TxFIFO (ep2.xact1 and ep1.xact2, in order).
11. The host repeats its attempt to read data (IN token) from endpoint 2.
12. Because endpoint 2's data is ready, the core responds with the data (ep2.xact_1), which the host ACKs.

Bit	Name	Reset	Access	Description
24:21	FN	0x0	R	**Frame Number device only**
	This is the least significant 4 bits of the Frame number in which the packet is received on the USB.			
20:17	PKTSTS	0x0	R	**Packet Status (host or device)**
	Indicates the status of the received packet.			

Value	Mode	Description
1	GOUTNAK	Device mode: Global OUT NAK (triggers an interrupt).
2	PKTRCV	Host mode: IN data packet received. Device mode: OUT data packet received.
3	XFERCOMPL	Host mode: IN transfer completed (triggers an interrupt). Device mode: OUT transfer completed (triggers an interrupt).
4	SETUPCOMPL	Device mode: SETUP transaction completed (triggers an interrupt).
5	TGLERR	Host mode: Data toggle error (triggers an interrupt).
6	SETUPRCV	Device mode: SETUP data packet received.
7	CHLT	Host mode: Channel halted (triggers an interrupt).

Bit	Name	Reset	Access	Description
16:15	DPID	0x0	R	**Data PID (host or device)**
	Host mode: Indicates the Data PID of the received packet. Device mode: Indicates the Data PID of the received OUT data packet.			

Value	Mode	Description
0	DATA0	DATA0 PID.
1	DATA1	DATA1 PID.
2	DATA2	DATA2 PID.
3	MDATA	MDATA PID.

Bit	Name	Reset	Access	Description
14:4	BCNT	0x000	R	**Byte Count (host or device)**
	Host mode: Indicates the byte count of the received IN data packet. Device mode: Indicates the byte count of the received data packet.			
3:0	CHEPNUM	0x0	R	**Channel Number host only / Endpoint Number device only**
	Host mode: Indicates the channel number to which the current received packet belongs. Device mode: Indicates the endpoint number to which the current received packet belongs.			

15.6.17 USB_GRXFSIZ - Receive FIFO Size Register

The application can program the RAM size that must be allocated to the RxFIFO.

Offset	Bit Position																													
0x3C024	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																							0x200								
Access																							RW								
Name																							RXFDEP								

Bit	Name	Reset	Access	Description
31:10	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
9:0	RXFDEP	0x200	RW	**RxFIFO Depth**
	This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 512.			

Bit	Name	Reset	Access	Description
	Value	Mode	Description	
	1	IN	Direction is IN.	
14:11	EPNUM	0x0	RW	**Endpoint Number**
	Indicates the endpoint number on the device serving as the data source or sink.			
10:0	MPS	0x000	RW	**Maximum Packet Size**
	Indicates the maximum packet size of the associated endpoint.			

15.6.36 USB_HCx_INT - Host Channel x Interrupt Register

This register indicates the status of a channel with respect to USB- and AHB-related events. The application must read this register when the Host Channels Interrupt bit of the Core Interrupt register (USB_GINTSTS.HCHINT) is set. Before the application can read this register, it must first read the Host All Channels Interrupt (USB_HAINT) register to get the exact channel number for the Host Channel x Interrupt register. The application must clear the appropriate bit in this register to clear the corresponding bits in the USB_HAINT and USB_GINTSTS registers.

Offset	Bit Position																													
0x3C508	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	0
Access																						RW1H	RW1H	RW1H	RW1H		RW1H	RW1H	RW1H	RW1H	RW1H	RW1H
Name																						DATATGLERR	FRMOVRUN	BBLERR	XACTERR		ACK	NAK	STALL	AHBERR	CHHLTD	XFERCOMPL

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	DATATGLERR	0	RW1H	**Data Toggle Error**
	This bit can be set only by the core and the application should write 1 to clear it.			
9	FRMOVRUN	0	RW1H	**Frame Overrun**
	This bit can be set only by the core and the application should write 1 to clear it.			
8	BBLERR	0	RW1H	**Babble Error**
	This bit can be set only by the core and the application should write 1 to clear it.			
7	XACTERR	0	RW1H	**Transaction Error**
	Indicates one of the following errors occurred on the USB: CRC check failure, Timeout, Bit stuff error or False EOP. This bit can be set only by the core and the application should write 1 to clear it.			
6	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
5	ACK	0	RW1H	**ACK Response Received/Transmitted Interrupt**
	This bit can be set only by the core and the application should write 1 to clear it.			
4	NAK	0	RW1H	**NAK Response Received Interrupt**
	This bit can be set only by the core and the application should write 1 to clear it.			
3	STALL	0	RW1H	**STALL Response Received Interrupt**
	This bit can be set only by the core and the application should write 1 to clear it.			
2	AHBERR	0	RW1H	**AHB Error**
	This is generated only in DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address.			

Many slave-only devices operating on an I2C-bus are not capable of driving SCL low, but in the rare case that SCL is stuck LOW, the advice is to apply a hardware reset signal to the slaves on the bus. If this does not work, cycle the power to the devices in order to make them release SCL.

When SDA is stuck low and SCL is free, a master should send 9 clock pulses on SCL while tristating the SDA. This procedure is performed in the GPIO module after clearing the I2C_ROUTE register and disabling the I2C module. The device that held the bus low should release it sometime within those 9 clocks. If not, use the same approach as for when SCL is stuck, resetting and possibly cycling power to the slaves.

Lockup of SDA can be detected by keeping count of the number of continuous arbitration losses during address transmission. If arbitration is also lost during the transmission of a general call address, i.e. during the transmission of the STOP condition, which should never happen during normal operation, this is a good indication of SDA lockup.

Detection of SCL lockups can be done using the timeout functionality defined in Section 16.3.12.6 (p. 435)

16.3.12.5 Bus Idle Timeout

When SCL has been high for a significant amount of time, this is a good indication of that the bus is idle. On an SMBus system, the bus is only allowed to be in this state for a maximum of 50 µs before the bus is considered idle.

The bus idle timeout BITO in I2Cn_CTRL can be used to detect situations where the bus goes idle in the middle of a transmission. The timeout can be configured in BITO, and when the bus has been idle for the given amount of time, the BITO interrupt flag in I2Cn_IF is set. The bus can also be set idle automatically on a bus idle timeout. This is enabled by setting GIBITO in I2Cn_CTRL.

When the bus idle timer times out, it wraps around and continues counting as long as its condition is true. If the bus is not set idle using GIBITO or the ABORT command in I2Cn_CMD, this will result in periodic timeouts.

Note

This timeout will be generated even if SDA is held low.

The bus idle timeout is active as long as the bus is busy, i.e. BUSY in I2Cn_STATUS is set. The timeout can be used to get the I2C module out of the busy-state it enters when reset, see Section 16.3.7.3 (p. 425) .

16.3.12.6 Clock Low Timeout

The clock timeout, which can be configured in CLTO in I2Cn_CTRL, starts counting whenever SCL goes low, and times out if SCL does not go high within the configured timeout. A clock low timeout results in CLTOIF in I2Cn_IF being set, allowing software to take action.

When the timer times out, it wraps around and continues counting as long as SCL is low. An SCL lockup will thus result in periodic clock low timeouts as long as SCL is low.

16.3.13 DMA Support

The I2C module has full DMA support. The DMA controller can write to the transmit buffer using the I2Cn_TXDATA register, and it can read from the receive buffer using the RXDATA register. A request for the DMA controller to read from the I2C receive buffer can come from the following source:

- Data available in the receive buffer

A write request can come from one of the following sources:

Bit	Name	Reset	Access	Description

When set, the bus automatically goes idle on a bus idle timeout, allowing new transfers to be initiated.

Value	Description
0	A bus idle timeout has no effect on the bus state.
1	A bus idle timeout tells the I^2C module that the bus is idle, allowing new transfers to be initiated.

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

13:12 BITO 0x0 RW **Bus Idle Timeout**

Use to generate a timeout when SCL has been high for a given amount time between a START and STOP condition. When in a bus transaction, i.e. the BUSY flag is set, a timer is started whenever SCL goes high. When the timer reaches the value defined by BITO, it sets the BITO interrupt flag. The BITO interrupt flag will then be set periodically as long as SCL remains high. The bus idle timeout is active as long as BUSY is set. It is thus stopped automatically on a timeout if GIBITO is set. It is also stopped a STOP condition is detected and when the ABORT command is issued. The timeout is activated whenever the bus goes BUSY, i.e. a START condition is detected.

Value	Mode	Description
0	OFF	Timeout disabled
1	40PCC	Timeout after 40 prescaled clock cycles. In standard mode at 100 kHz, this results in a 50us timeout.
2	80PCC	Timeout after 80 prescaled clock cycles. In standard mode at 100 kHz, this results in a 100us timeout.
3	160PCC	Timeout after 160 prescaled clock cycles. In standard mode at 100 kHz, this results in a 200us timeout.

11:10 Reserved — To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)

9:8 CLHR 0x0 RW **Clock Low High Ratio**

Determines the ratio between the low and high parts of the clock signal generated on SCL as master.

Value	Mode	Description
0	STANDARD	The ratio between low period and high period counters (N_{low}:N_{high}) is 4:4
1	ASYMMETRIC	The ratio between low period and high period counters (N_{low}:N_{high}) is 6:3
2	FAST	The ratio between low period and high period counters (N_{low}:N_{high}) is 11:6

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

6 GCAMEN 0 RW **General Call Address Match Enable**

Set to enable address match on general call in addition to the programmed slave address.

Value	Description
0	General call address will be NACK'ed if it is not included by the slave address and address mask.
1	When a general call address is received, a software response is required.

5 ARBDIS 0 RW **Arbitration Disable**

A master or slave will not release the bus upon losing arbitration.

Value	Description
0	When a device loses arbitration, the ARB interrupt flag is set and the bus is released.
1	When a device loses arbitration, the ARB interrupt flag is set, but communication proceeds.

4 AUTOSN 0 RW **Automatic STOP on NACK**

Write to 1 to make a master transmitter send a STOP when a NACK is received from a slave.

Value	Description
0	Stop is not automatically sent if a NACK is received from a slave.
1	The master automatically sends a STOP if a NACK is received from a slave.

3 AUTOSE 0 RW **Automatic STOP when Empty**

Write to 1 to make a master transmitter send a STOP when no more data is available for transmission.

Value	Description
0	A stop must be sent manually when no more data is to be transmitted.
1	The master automatically sends a STOP when no more data is available for transmission.

2 AUTOACK 0 RW **Automatic Acknowledge**

Set to enable automatic acknowledges.

Bit	Name	Reset	Access	Description
1	RXDIS	0	W1	**Receiver Disable**
	Set to disable data reception. If a frame is under reception when the receiver is disabled, the incoming frame is discarded.			
0	RXEN	0	W1	**Receiver Enable**
	Set to activate data reception on LEUn_RX.			

19.5.3 LEUARTn_STATUS - Status 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	1	0	0	0	0
Access																											R	R	R	R	R	R
Name																											RXDATAV	TXBL	TXC	RXBLOCK	TXENS	RXENS

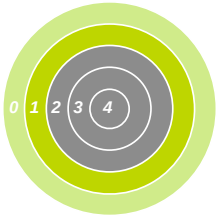
Bit	Name	Reset	Access	Description
31:6	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
5	RXDATAV	0	R	**RX Data Valid**
	Set when data is available in the receive buffer. Cleared when the receive buffer is empty.			
4	TXBL	1	R	**TX Buffer Level**
	Indicates the level of the transmit buffer. Set when the transmit buffer is empty, and cleared when it is full.			
3	TXC	0	R	**TX Complete**
	Set when a transmission has completed and no more data is available in the transmit buffer. Cleared when a new transmission starts.			
2	RXBLOCK	0	R	**Block Incoming Data**
	When set, the receiver discards incoming frames. An incoming frame will not be loaded into the receive buffer if this bit is set at the instant the frame has been completely received.			
1	TXENS	0	R	**Transmitter Enable Status**
	Set when the transmitter is enabled.			
0	RXENS	0	R	**Receiver Enable Status**
	Set when the receiver is enabled. The receiver must be enabled for start frames, signal frames, and multi-processor address bit detection.			

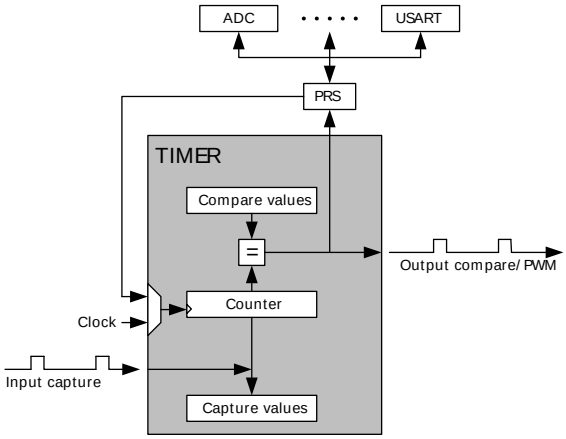
19.5.4 LEUARTn_CLKDIV - Clock Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 20) .

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																		0x000														
Access																		RW														
Name																		DIV														

20 TIMER - Timer/Counter





Quick Facts

What?

The TIMER (Timer/Counter) keeps track of timing and counts events, generates output waveforms and triggers timed actions in other peripherals.

Why?

Most applications have activities that need to be timed accurately with as little CPU intervention and energy consumption as possible.

How?

The flexible 16-bit TIMER can be configured to provide PWM waveforms with optional dead-time insertion for e.g. motor control, or work as a frequency generator. The Timer can also count events and control other peripherals through the PRS, which offloads the CPU and reduce energy consumption.

20.1 Introduction

The 16-bit general purpose Timer has 3 compare/capture channels for input capture and compare/Pulse-Width Modulation (PWM) output. TIMER0 also includes a Dead-Time Insertion module suitable for motor control applications.

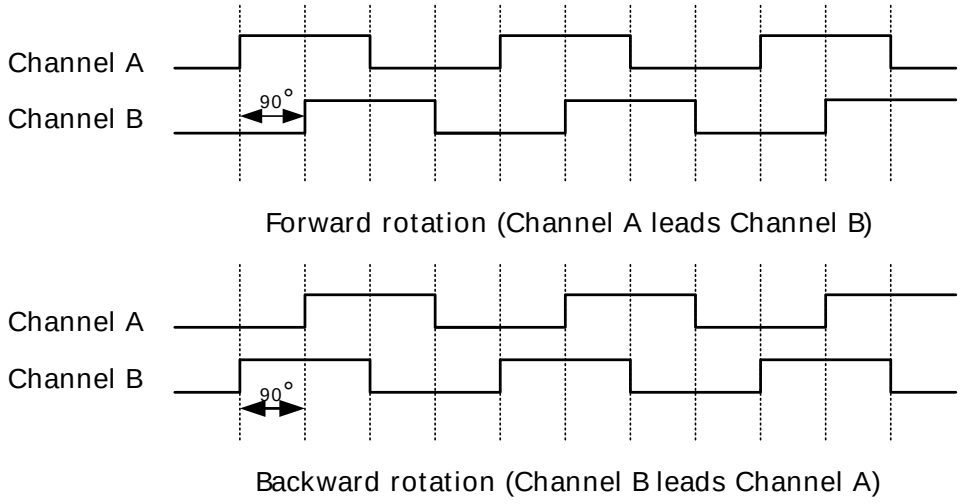
20.2 Features

- 16-bit auto reload up/down counter
  - Dedicated 16-bit reload register which serves as counter maximum
- 3 Compare/Capture channels
  - Individual configurable as either input capture or output compare/PWM
- Multiple Counter modes
  - Count up
  - Count down
  - Count up/down
  - Quadrature Decoder
  - Direction and count from external pins
- 2x Count Mode
- Counter control from PRS or external pin
  - Start
  - Stop
  - Reload and start
- Inter-Timer connection
  - Allows 32-bit counter mode
  - Start/stop synchronization between several Timers
- Input Capture

20.3.1.6 Quadrature Decoder

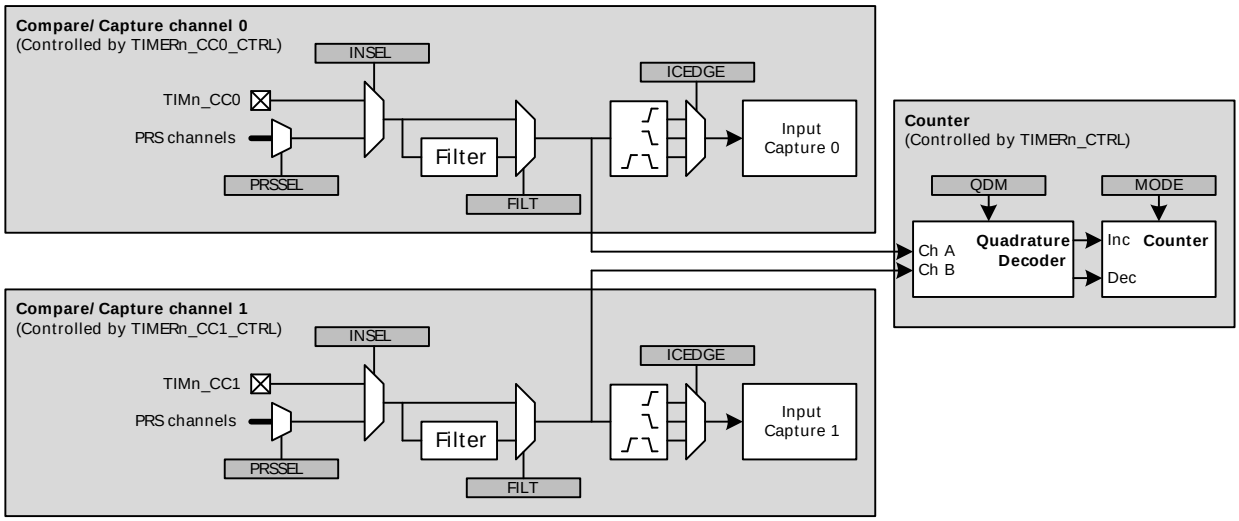
Quadrature Decoding mode is used to track motion and determine both rotation direction and position. The Quadrature Decoder uses two input channels that are 90 degrees out of phase (see Figure 20.6 (p. 528)).

Figure 20.6. TIMER Quadrature Encoded Inputs



In the Timer these inputs are tapped from the Compare/Capture channel 0 (Channel A) and 1 (Channel B) inputs before edge detection. The Timer/Counter then increments or decrements the counter, based on the phase relation between the two inputs. The Quadrature Decoder Mode supports two channels, but if a third channel (Z-terminal) is available, this can be connected to an external interrupt and trigger a counter reset from the interrupt service routine. By connecting a periodic signal from another timer as input capture on Compare/Capture Channel 2, it is also possible to calculate speed and acceleration.

Figure 20.7. TIMER Quadrature Decoder Configuration



The Quadrature Decoder can be set in either X2 or X4 mode, which is configured in the QDM bit in TIMERn_CTRL. See Figure 20.7 (p. 528)

continue running if triggered while it is running, so the multiple-triggering will only have an effect if you try to disable the RTC when it is being triggered.

23.3.3.5 Debug

If DEBUGRUN in LETIMERn_CTRL is cleared, the LETIMER automatically stops counting when the CPU is halted during a debug session, and resumes operation when the CPU continues. Because of synchronization, the LETIMER is halted two clock cycles after the CPU is halted, and continues running two clock cycles after the CPU continues. RUNNING in LETIMERn_STATUS is not cleared when the LETIMER stops because of a debug-session.

Set DEBUGRUN in LETIMERn_CTRL to allow the LETIMER to continue counting even when the CPU is halted in debug mode.

23.3.4 Underflow Output Action

For each of the repeat registers, an underflow output action can be set. The configured output action is performed every time the counter underflows while the respective repeat register is nonzero. In PWM mode, the output is similarly only changed on COMP1 match if the repeat register is nonzero. As an example, the timer will perform 7 output actions if LETIMERn_REP0 is set to 7 when starting the timer in one-shot mode and leaving it untouched for a while.

The output actions can be set by configuring UFOA0 and UFOA1 in LETIMERn_CTRL. UFOA0 defines the action on output 0, and is connected to LETIMERn_REP0, while UFOA1 defines the action on output 1 and is connected to LETIMERn_REP1. The possible actions are defined in Table 23.2 (p. 592) .

Table 23.2. LETIMER Underflow Output Actions

UF0A0/UF0A1	Mode	Description
00	Idle	The output is held at its idle value
01	Toggle	The output is toggled on LETIMERn_CNT underflow if LEIMERn_REPx is nonzero
10	Pulse	The output is held active for one clock cycle on LETIMERn_CNT underflow if LETIMERn_REPx is nonzero. It then returns to its idle value
11	PWM	The output is set idle on LETIMERn_CNT underflow and active on compare match with LETIMERn_COMP1 if LETIMERn_REPx is nonzero.

Note

For the Pulse and PWM modes, the outputs will return to their idle states regardless of the state of the corresponding LETIMERn_REPx registers. They will only be set active if the LETIMERn_REPx registers are nonzero however.

The polarity of the outputs can be set individually by configuring OPOL0 and OPOL1 in LETIMERn_CTRL. When these are cleared, their respective outputs have a low idle value and a high active value. When they are set, the idle value is high, and the active value is low.

When using the toggle action, the outputs can be driven to their idle values by setting their respective CTO0/CTO1 command bits in LETIMERn_CTRL. This can be used to put the output in a well-defined state before beginning to generate toggle output, which may be important in some applications. The command bit can also be used while the timer is running.

As the auxiliary counter, the main counter can be configured to count only on certain events. This is done through CNTEV in PCNTn_CTRL, and it is possible like for the auxiliary counter, to make the main counter count on only up and down events. The difference between the counters is that where the auxiliary counter will only count up, the main counter will count up or down depending on the direction of the count event.

24.3.4 Register Access

The counter-clock domain may be clocked externally. To update the counter-clock domain registers from software in this mode, 2-3 clock pulses on the external clock are needed to synchronize accesses to the externally clocked domain. Clock source switching is controlled from the registers in the CMU (Chapter 11 (p. 126)).

When the RSTEN bit in the PCNTn_CTRL register is set to 1, the PCNT clock domain is asynchronously held in reset. The reset is synchronously released two PCNT clock edges after the RSTEN bit in the PCNTn_CTRL register is cleared by software. This asynchronous reset restores the reset values in PCNTn_TOP, PCNTn_CNT and other control registers in the PCNT clock domain.

Since this module is a Low Energy Peripheral, and runs off a clock which is asynchronous to the HFCORECLK, special considerations must be taken when accessing registers. Please refer to Section 5.3 (p. 20) for a description on how to perform register accesses to Low Energy Peripherals.

Note

PCNTn_TOP and PCNTn_CNT are read-only registers. When writing to PCNTn_TOPB, make sure that the counter value, PCNTn_CNT, can not exceed the value written to PCNTn_TOPB within two clock cycles.

24.3.5 Clock Sources

The 32 kHz LFACLK is one of two possible clock sources. The clock select register is described in Chapter 11 (p. 126) . The default clock source is the LFACLK.

This PCNT module may also use PCNTn_S0IN as an external clock to clock the counter (EXTCLKSINGLE mode) and to sample PCNTn_S1IN (EXTCLKQUAD mode). Setup, hold and max frequency constraints for PCNTn_S0IN and PCNTn_S1IN for these modes are specified in the device datasheet.

To use this module, the LE interface clock must be enabled in CMU_HFCORECLKEN0, in addition to the module clock.

Note

PCNT Clock Domain Reset, RSTEN, should be set when changing clock source for PCNT. In addition to this, the PCNTn_SYNCBUSY value should be zero. If changing to an external clock source, the clock pin has to be enabled as input prior to de-asserting RSTEN. Changing clock source without asserting RSTEN results in undefined behaviour.

24.3.6 Input Filter

An optional pulse width filter is available in OVSSINGLE mode. The filter is enabled by writing 1 to the FILT bit in the PCNTn_CTRL register. When enabled, the high and low periods of PCNTn_S0IN must be stable for 5 consecutive clock cycles before the edge is passed to the edge detector.

In EXTCLKSINGLE and EXTCLKQUAD mode, there is no digital pulse width filter available.

24.3.7 Edge Polarity

The edge polarity can be set by configuring the EDGE bit in the PCNTn_CTRL register. When this bit is cleared, the pulse counter counts positive edges in OVSSINGLE mode and negative edges if the bit is set.

Bit	Name	Reset	Access	Description

Value	Mode	Description
0	4CYCLES	4 HFPERCLK cycles.
1	8CYCLES	8 HFPERCLK cycles.
2	16CYCLES	16 HFPERCLK cycles.
3	32CYCLES	32 HFPERCLK cycles.
4	64CYCLES	64 HFPERCLK cycles.
5	128CYCLES	128 HFPERCLK cycles.
6	256CYCLES	256 HFPERCLK cycles.
7	512CYCLES	512 HFPERCLK cycles.

7	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
6:4	HYSTSEL	0x0	RW	**Hysteresis Select**

Select hysteresis level. The hysteresis levels can vary, please see the electrical characteristics for the device for more information.

Value	Mode	Description
0	HYST0	No hysteresis.
1	HYST1	~15 mV hysteresis.
2	HYST2	~22 mV hysteresis.
3	HYST3	~29 mV hysteresis.
4	HYST4	~36 mV hysteresis.
5	HYST5	~43 mV hysteresis.
6	HYST6	~50 mV hysteresis.
7	HYST7	~57 mV hysteresis.

3	GPIOINV	0	RW	**Comparator GPIO Output Invert**

Set this bit to 1 to invert the comparator alternate function output to GPIO.

Value	Mode	Description
0	NOTINV	The comparator output to GPIO is not inverted.
1	INV	The comparator output to GPIO is inverted.

2	INACTVAL	0	RW	**Inactive Value**

The value of this bit is used as the comparator output when the comparator is inactive.

Value	Mode	Description
0	LOW	The inactive value is 0.
1	HIGH	The inactive state is 1.

1	MUXEN	0	RW	**Input Mux Enable**

Enable Input Mux. Setting the EN bit will also enable the input mux.

0	EN	0	RW	**Analog Comparator Enable**

Enable/disable analog comparator.

26.5.2 ACMPn_INPUTSEL - Input Selection 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			0x0					0								1			0x00						0x8					0x0		
Access			RW					RW								RW			RW						RW					RW		
Name			CSRESSEL					CSRESEN								LPREF			VDDLEVEL						NEGSEL					POSSEL		

Bit	Name	Reset	Access	Description
31:30	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 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	WARMUP	0	R	**Warm-up Interrupt Flag**
	Indicates that the analog comparator warm-up period is finished.			
0	EDGE	0	R	**Edge Triggered Interrupt Flag**
	Indicates that there has been a rising or falling edge on the analog comparator output.			

26.5.6 ACMPn_IFS - Interrupt Flag Set Register

Offset	Bit Position																													
0x014	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																															W1	W1
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	W1	**Warm-up Interrupt Flag Set**
	Write to 1 to set warm-up finished interrupt flag.			
0	EDGE	0	W1	**Edge Triggered Interrupt Flag Set**
	Write to 1 to set edge triggered interrupt flag.			

26.5.7 ACMPn_IFC - Interrupt Flag Clear Register

Offset	Bit Position																													
0x018	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																															W1	W1
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	W1	**Warm-up Interrupt Flag Clear**
	Write to 1 to clear warm-up finished interrupt flag.			
0	EDGE	0	W1	**Edge Triggered Interrupt Flag Clear**
	Write to 1 to clear edge triggered interrupt flag.			

28.4 Register Map

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

Offset	Name	Type	Description
0x000	ADCn_CTRL	RW	Control Register
0x004	ADCn_CMD	W1	Command Register
0x008	ADCn_STATUS	R	Status Register
0x00C	ADCn_SINGLECTRL	RW	Single Sample Control Register
0x010	ADCn_SCANCTRL	RW	Scan Control Register
0x014	ADCn_IEN	RW	Interrupt Enable Register
0x018	ADCn_IF	R	Interrupt Flag Register
0x01C	ADCn_IFS	W1	Interrupt Flag Set Register
0x020	ADCn_IFC	W1	Interrupt Flag Clear Register
0x024	ADCn_SINGLEDATA	R	Single Conversion Result Data
0x028	ADCn_SCANDATA	R	Scan Conversion Result Data
0x02C	ADCn_SINGLEDATAP	R	Single Conversion Result Data Peek Register
0x030	ADCn_SCANDATAP	R	Scan Sequence Result Data Peek Register
0x034	ADCn_CAL	RW	Calibration Register
0x03C	ADCn_BIASPROG	RW	Bias Programming Register

28.5 Register Description

28.5.1 ADCn_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					0x0							0x1F						0x00									0x0		0		0x0	
Access					RW							RW						RW									RW		RW		RW	
Name					OVSRSEL							TIMEBASE						PRESC									LPFMODE		TAILGATE		WARMUPMODE	

Bit	Name	Reset	Access	Description
31:28	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
27:24	OVSRSEL	0x0	RW	**Oversample Rate Select**

Select oversampling rate. Oversampling must be enabled for each mode for this setting to take effect.

Value	Mode	Description
0	X2	2 samples for each conversion result
1	X4	4 samples for each conversion result
2	X8	8 samples for each conversion result
3	X16	16 samples for each conversion result
4	X32	32 samples for each conversion result
5	X64	64 samples for each conversion result
6	X128	128 samples for each conversion result
7	X256	256 samples for each conversion result
8	X512	512 samples for each conversion result

30.3.2.5 Cascaded Inverting PGA

This mode enables the opamp signals to be internally configured to cascade two or three opamps in inverting mode as shown in Figure 30.7 (p. 739) . In both cases the positive input will be configured to signal ground by setting OPAxPOSSEL bit-field to PAD in DACn_OPAx_MUX. When cascaded, the negative input is connected to the resistor ladder by setting the OPAxNEGSEL bit-field to OPATAP in DACn_OPAxMUX. The input to the resistor ladder can be configured in the OPAxRESINMUX bit-field in DAC_nOPAxMUX. The output from OPA0 can be connected to OPA1 to create the second stage by setting the NEXTOUT bit-field in DACn_OPAxMUX. To complete the stage, OPA1RESINMUX field must be set to OPA0INP. Similarly, the last stage can be created by setting the NEXTOUT bit-field in DACn_OPA1MUX and OPA2RESINMUX bit-field to OPA1INP in DACn_OPA2MUX.

Figure 30.7. Cascaded Inverting PGA Overview

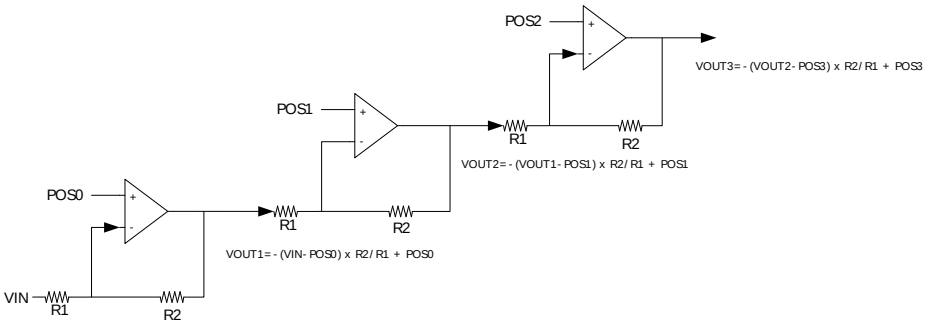


Table 30.5. Cascaded Inverting PGA Configuration

OPA	OPA bit-fields	OPA Configuration
OPA0	POSSEL	POSPAD0
OPA0	NEGSEL	OPA0TAP
OPA0	RESINMUX	NEGPAD0
OPA0	NEXTOUT	1
OPA1	POSSEL	POSPAD1
OPA1	NEGSEL	OPATAP
OPA1	RESINMUX	OPA0INP
OPA1	NEXTOUT	1
OPA2	POSSEL	POSPAD2
OPA2	NEGSEL	OPATAP
OPA2	RESINMUX	OPA1INP

30.3.2.6 Cascaded Non-inverting PGA

This mode enables the opamp signals to be internally configured to cascade two or three opamps in non-inverting mode as shown in Figure 30.8 (p. 740) . In both cases the negative input for all opamps will be connected to the resistor ladder by setting the OPAxNEGSEL bit-field to OPATAP. In addition the resistor ladder input must be set to VSS or NEGPADx in the OPAxRESINMUX in DACn_OPAxMUX. When cascaded, the positive input on OPA0 is configured by the OPA0POSSEL bit-field. The output from OPA0 can be connected to OPA1 to create the second stage by setting NEXTOUT in DACn_OPA0MUX. To complete the stage, the OPA1POSSEL bit-field must be set to OPA0INP in DACn_OPA1MUX. Similarly,

Bit	Name	Reset	Access	Description

Select input port for external interrupt 8.

Value	Mode	Description
0	PORTA	Port A pin 8 selected for external interrupt 8
1	PORTB	Port B pin 8 selected for external interrupt 8
2	PORTC	Port C pin 8 selected for external interrupt 8
3	PORTD	Port D pin 8 selected for external interrupt 8
4	PORTE	Port E pin 8 selected for external interrupt 8
5	PORTF	Port F pin 8 selected for external interrupt 8

32.5.12 GPIO_EXTIRISE - External Interrupt Rising Edge Trigger Register

Offset	Bit Position																															
0x108	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																	0x0000															
Access																	RW															
Name																	EXTIRISE															

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15:0	EXTIRISE	0x0000	RW	**External Interrupt n Rising Edge Trigger Enable**

Set bit n to enable triggering of external interrupt n on rising edge.

Value	Description
EXTIRISE[n] = 0	Rising edge trigger disabled
EXTIRISE[n] = 1	Rising edge trigger enabled

32.5.13 GPIO_EXTIFALL - External Interrupt Falling Edge Trigger Register

Offset	Bit Position																															
0x10C	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																	0x0000															
Access																	RW															
Name																	EXTIFALL															

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15:0	EXTIFALL	0x0000	RW	**External Interrupt n Falling Edge Trigger Enable**

cleared when LCD_AREGA or LCD_AREGB are updated with new values. See Table 33.15 (p. 804) for an example.

Table 33.15. LCD Animation Example

ASTATE	LCD_AREGA	LCD_AREGB	Resulting Data
0	11000000	11000000	11000000
1	01100000	11000000	11100000
2	01100000	01100000	01100000
3	00110000	01100000	01110000
4	00110000	00110000	00110000
5	00011000	00110000	00111000
6	00011000	00011000	00011000
7	00001100	00011000	00011100
8	00001100	00001100	00001100
9	00000110	00001100	00001110
10	00000110	00000110	00000110
11	00000011	00000110	00000111
12	00000011	00000011	00000011
13	10000001	00000011	10000011
14	10000001	10000001	10000001
15	11000000	10000001	11000001

In the table, AREGASC = 10, AREGBSC = 10, ALOGSEL = 1 and the resulting data is to be displayed on segment lines 7-0 or 15-8 multiplexed with LCD_COM0.

Figure 33.44. LCD Block Diagram of the Animation Circuit