

UCD30xx Faults and External Interrupts **(GIO) Programmer's Manual**

Table of Contents

Table of Contents	2
1 Overview	3
2 Fault signal timing	4
3 Register Map	5
3.1 Fault IO Direction Register (FAULTDIR)	5
3.2 Fault IN Register (FAULTIN)	5
3.3 Fault Out Register (FAULTOUT)	6
3.4 Fault Set Ports Register (FAULTSET)	6
3.5 Fault Clear Ports Register (FAULTCLR)	6
3.6 Fault Interrupt Enable Register (FAULTINTENA)	6
3.7 Fault Interrupt Polarity Register (FAULTINTPOL)	7
3.8 Fault Interrupt Pending Register (FAULTINTPEND)	8
3.9 EXT_INT IO Direction Register (EXTINTDIR)	8
3.10 EXT_INT IN Register (EXTINTIN)	9
3.11 EXT_INT OUT Register (EXTINTOUT)	9
3.12 EXT_INT Set Ports Register (EXTINTSET)	10
3.13 EXT_INT Clear Ports Register (EXTINTCLR)	10
1.14 EXT_INT Interrupt Enable Register (EXTINTINTENA)	10
1.15 EXT_INT Interrupt Polarity Register (EXTINTINTPOL)	10
1.16 EXT_INT Interrupt Pending Register (EXTINTINTPEND)	11
4 Useful C statements	12

1 Overview

The General Purpose Input-Output/Fault (GPIO_FAULT) ports are for pins that are not associated with any communication port. These Bi-directional pins can be configured, by firmware, as output or input pins. When configured as outputs, the pins can be set to output a “1” or “0” value by writing to the appropriate output register. When configured as inputs, the pin status i.e., the digital value at the pin inputs, can be read through memory map reads of the appropriate input register. Two of the pins, INT1 and INT2, have additional external interrupt capability. These interrupts can be configured for either falling or rising edge detection. Interrupts can be enabled or disabled and flags can be monitored for level status.

GPIO_FAULT Registers have the following attributes:

- One 8-bit wide FAULT port
- One 2-bit wide EXT_INT port
- Addresses placed on word boundaries
- Byte, Half-word and Word Writes are permitted
- All Registers can be read in any mode
- All Registers are writeable

The fault pins can be activated to turn off a specific DPWM output without any firmware intervention. The logical connection between a fault input and a DPWM output is hard-wired and is not configurable.

After the DPWM output is turned off by a fault input it stays latched off even if fault pin toggles back to the normal state. Firmware acknowledge of the fault by clearing the interrupt pending flag is required in order to turn on the specific DPWM output. Please note a “1” needs to be written this bit in order to clear it.

Fault 1A is linked to turn off DPWM1 channel A, fault 1B turns off DPWM_1B, fault pin 4B will cause output B of DPWM4 to turn off.

If a DPWM is configured into “Multi output mode”, then the mapped fault input will turn off only the relevant DPWM channel (either A or B). In all other DPWM output modes of configuration (Normal mode, Phase shifting mode or Resonant mode), both channels A and B will be turned off by activation of either one of the fault pins that are mapped to any of the DPWM outputs.

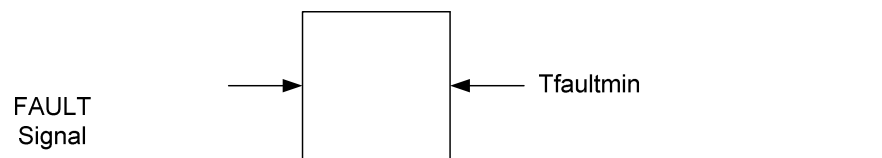
Fault outputs can be configured to invoke interrupts as well. This feature is specially useful if extra DPWM outputs need to be turned off. In this case the DPWM outputs that are not mapped to the specific fault input can be turned off by the relevant interrupt service routine.

2 Fault signal timing

The Fault pins are sampled on a clocked basis. The clock used is the ICLK, which has a nominal frequency of 15.625 MHz. See the datasheet for clock frequency variation.

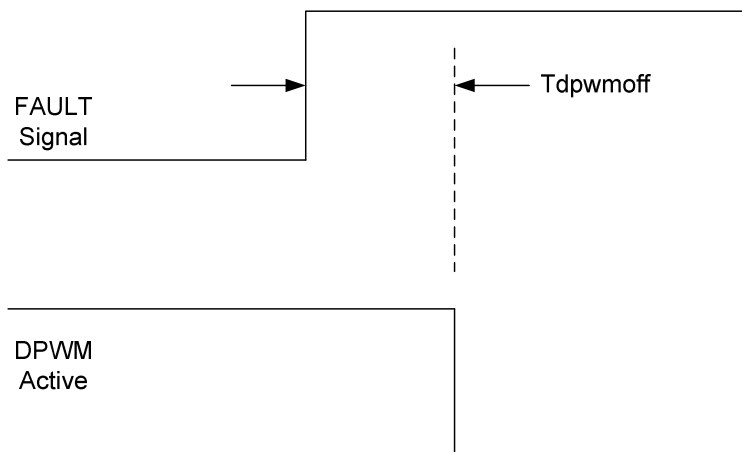
This sampling rate affects response time of the Fault pins, and also imposes a minimum pulse width for guaranteed detection of a fault event.

The minimum pulse width $T_{faultmin}$ must be at least one ICLK period for a fault to be detected.



Shorter pulses may be detected, depending on their timing relative to the ICLK, but this cannot be guaranteed.

The delay from the Fault signal going active to the DPWM turn off $T_{dpwmoff}$ will vary from 1 to 2 ICLKs plus a small propagation delay (see data sheet)



The Fault signal is sampled, and the sampling requires an edge. If the Fault signal is already high when the Fault input is enabled, it will have to go low (for at least 1 ICLK period) and then high again so that the internal logic can see an edge.

To compensate for this, firmware that enables the Fault signal may need to check just before and just after the enable to ensure that the Fault signal is still inactive at that time.

3 Register Map

3.1 Fault IO Direction Register (FAULTDIR)

Address FFF7FA00

Bit Number	7	6	5	4	3	2	1	0
Bit Name	DIR7	DIR6	DIR5	DIR4	DIR3	DIR2	DIR1	DIR0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bits 7-0: DIRn – FAULT[n] Controls the direction of the pins (n ranges from 0 to 7). When the pins associated with these 8-bits are used as DPWM FAULT input pins, then for n ranging from 0 to 7 corresponds to the pins FAULT_1A, FAULT1B, FAULT_2A, FAULT2B, FAULT_3A, FAULT3B, FAULT_4A and FAULT4B respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 7 corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = FAULT[n] configured as an input pin (Default)

1 = FAULT[n] configured as an output pin

3.2 Fault IN Register (FAULTIN)

Address FFF7FA04

Bit Number	7	6	5	4	3	2	1	0
Bit Name	IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
Access	R	R	R	R	R	R	R	R
Default	-	-	-	-	-	-	-	-

Bits 7-0: INn – FAULT[n] Pin input value (n ranges from 0 to 7). When the pins associated with these 8-bits are used as DPWM FAULT input pins, then for n ranging from 0 to 7 corresponds to the pins FAULT_1A, FAULT1B, FAULT_2A, FAULT2B, FAULT_3A, FAULT3B, FAULT_4A and FAULT4B respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 7 corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = FAULT[n] observed at logic level low
 1 = FAULT[n] observed at logic level high

3.3 Fault Out Register (FAULTOUT)

Address FFF7FA08

Bit Number	7	6	5	4	3	2	1	0
Bit Name	OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Bits 7-0: OUT_n – FAULT[n] Pin output value (n ranges from 0 to 7). These bits are used only when the 8 pins associated with these are configured as simple GPIO pins. Then for n ranging from 0 to 7, corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = FAULT[n] driven low when configured as output (Default)
 1 = FAULT[n] driven high when configured as output

3.4 Fault Set Ports Register (FAULTSET)

Address FFF7FA0C

Bit Number
Bit Name

Bit NA: A write to this Register sets all FAULT Port outputs (section 1.3) to 1.

3.5 Fault Clear Ports Register (FAULTCLR)

Address FFF7FA10

Bit Number
Bit Name

Bit NA: A write to this Register clears all FAULT Port outputs (section 1.3) to 0.

3.6 Fault Interrupt Enable Register (FAULTINTENA)

Address FFF7FA14

Bit Number	7	6	5	4
Bit Name	INTENA7	INTENA6	INTENA5	INTENA4
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bit Number	3	2	1	0
Bit Name	INTENA3	INTENA2	INTENA1	INTENA0
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bits 7-0: INTENAn – FAULT[n] Interrupt Enable (n ranges from 0 to 7).

When the pins associated with these 8-bits are used as DPWM FAULT input pins, then for n ranging from 0 to 7 corresponds to the pins FAULT_1A, FAULT1B, FAULT_2A, FAULT2B, FAULT_3A, FAULT3B, FAULT_4A and FAULT4B respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 7 corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = Interrupt disabled for FAULT[n] pin (Default)

1 = Interrupt enabled for FAULT[n] pin

3.7 Fault Interrupt Polarity Register (FAULTINTPOL)

Address FFF7FA18

Bit Number	7	6	5	4
Bit Name	INTPOL7	INTPOL6	INTPOL5	INTPOL4
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bit Number	3	2	1	0
Bit Name	INTPOL3	INTPOL2	INTPOL1	INTPOL0
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bits 7-0: INTPOLn – FAULT[n] Interrupt Polarity Select (n ranges from 0 to 7).

When the pins associated with these 8-bits are used as DPWM FAULT input pins, then for n ranging from 0 to 7 corresponds to the pins FAULT_1A, FAULT1B, FAULT_2A, FAULT2B, FAULT_3A, FAULT3B, FAULT_4A and FAULT4B respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 7 corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = Interrupt generated on falling edge (Default)
 1 = Interrupt generated on rising edge

3.8 Fault Interrupt Pending Register (FAULTINTPEND)

Address FFF7FA1C

Bit Number	7	6	5	4
Bit Name	INTPEND7	INTPEND6	INTPEND5	INTPEND4
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bit Number	3	2	1	0
Bit Name	INTPEND3	INTPEND2	INTPEND1	INTPEND0
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bits 7-0: INTPENDn – This register indicates which FAULT[n] port has caused an interrupt. Writing a 1 to a bit will clear the interrupt flag. When the pins associated with these 8-bits are used as DPWM FAULT input pins, then for n ranging from 0 to 7 corresponds to the pins FAULT_1A, FAULT1B, FAULT_2A, FAULT2B, FAULT_3A, FAULT3B, FAULT_4A and FAULT4B respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 7 corresponds to the pins GPIO_08, GPIO_09, GPIO_10, GPIO_11, GPIO_30, GPIO_31, GPIO_32 and GPIO_33 respectively.

0 = No Interrupt detected (Default)
 1 = Interrupt pending

3.9 EXT_INT IO Direction Register (EXTINTDIR)

Address FFF7FA20

Bit Number	1	0
Bit Name	DIR1	DIR0
Access	R/W	R/W
Default	0	0

Formatted Table

Bits 1-0: DIR_n – EXT_INT[n] Controls the direction of the pins (n ranges from 0 to 1). When the pins associated with these 2-bits are used as external interrupt pins, then for n ranging from 0 to 1 corresponds to the pins INT1, INT2 respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

Deleted: 7

Deleted: 7

0 = EXT_INT[n] pin configured as an input (Default)
 1 = EXT_INT[n] pin configured as an output

3.10 EXT_INT IN Register (EXTINTIN)

Address FFF7FA24

Bit Number	1	0
Bit Name	IN1	IN0
Access	R	R
Default	-	-

Formatted Table

Bits 1-0: IN_n – EXT_INT[n] Pin input value (n ranges from 0 to 1). When the pins associated with these 2-bits are used as external interrupt pins, then for n ranging from 0 to 1 corresponds to the pins INT1, INT2 respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

Deleted: 7

Deleted: 7

0 = EXT_INT[n] observed at logic level low
 1 = EXT_INT[n] observed at logic level high

3.11 EXT_INT OUT Register (EXTINTOUT)

Address FFF7FA28

Bit Number	1	0
Bit Name	OUT1	OUT0
Access	R/W	R/W
Default	0	0

Formatted Table

Bits 7-0: OUT_n – EXT_INT[n] Pin output value (n ranges from 0 to 7). These bits are used when the pins associated with these are used as simple GPIO pins. Then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

0 = EXT_INT[n] driven low when configured as output (Default)
1 = EXT_INT[n] driven high when configured as output

3.12 EXT_INT Set Ports Register (EXTINTSET)

Address FFF7FA2C

Bit Number
Bit Name

Bit NA: A write to this Register sets all EXT_INT Port outputs (section 1.11) to 1.

3.13 EXT_INT Clear Ports Register (EXTINTCLR)

Address FFF7FA30

Bit Number
Bit Name

Bit NA: A write to this Register clears all EXT_INT Port outputs (section 1.11) to 0.

1.14 EXT_INT Interrupt Enable Register (EXTINTINTENA)

Address FFF7FA34

Bit Number	1	0
Bit Name	INTENA1	INTENA0
Access	R/W	R/W
Default	0	0

Deleted: ¶

Formatted: Left, Don't keep with next, Don't keep lines together

Formatted Table

Bits 1-0: INTENAn – EXT_INT[n] Interrupt Enable (n ranges from 0 to 1). When the pins associated with these 2-bits are used as external interrupt pins, then for n ranging from 0 to 1 corresponds to the pins INT1, INT2 respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

Deleted: 7

Deleted: 7

0 = Interrupt disabled for EXT_INT[n] pin (Default)
1 = Interrupt enabled for EXT_INT[n] pin

1.15 EXT_INT Interrupt Polarity Register (EXTINTINTPOL)

Address FFF7FA38

Formatted: Left, Don't keep with next, Don't keep lines together

Deleted: ¶

Bit Number	1	0
Bit Name	INTPOL1	INTPOL0
Access	R/W	R/W
Default	0	0

Formatted Table

Bits 1-0: INTPOLn – EXT_INT[n] Interrupt Polarity Select (n ranges from 0 to 1). When the pins associated with these 2-bits are used as external interrupt pins, then for n ranging from 0 to 1 corresponds to the pins INT1, INT2 respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

Deleted: 7

Deleted: 7

0 = Interrupt generated on falling edge (Default)
 1 = Interrupt generated on rising edge

1.16 **EXT_INT Interrupt Pending Register (EXTINTINTPEND)**

Address **FFF7FA3C**

Bit Number	1	0
Bit Name	INTPEND1	INTPEND0
Access	R/W	R/W
Default	0	0

Deleted: ¶
 -----Page Break-----
EXT_INT Interrupt Pending Register (EXTINTINTPEND)¶
 Formatted: Left, Don't keep with next, Don't keep lines together
 Deleted: ¶
 Formatted Table
 Deleted: 7

Bits 1-0: INTPENDn – This register indicates which EXT_INT[n] port has caused an interrupt. Writing a 1 to a bit will clear the interrupt flag. When the pins associated with these 2-bits are used as external interrupt pins, then for n ranging from 0 to 1 corresponds to the pins INT1, INT2 respectively. When the pins are used as simple GPIO pins, then for n ranging from 0 to 1 correspond to the pins GPIO_25 and GPIO_27 respectively.

0 = No Interrupt detected (Default)
 1 = Interrupt pending

4 Useful C statements

```
GioRegs.FAULTDIR.bytes.BYTE0 = 0;  
//Means: Set all fault pins as inputs
```

```
Dpwm4Regs.DPWMCTRL1.bit.FAULT_ENA = 0;  
// Means: Don't Shut DPWM4 output off even when the relevant fault input is asserted
```

```
GioRegs.EXTINTINTENA.bit.INTENA7 = 1;  
// Means: Set the last(8th) fault pin to generate an interrupt when asserted
```

```
GioRegs.EXTINTINTPOL.bit.INTPOL7 = 1;  
// Means: Generate interrupt on rising edge of the last(8th) fault pin (if interrupt already enabled)
```

```
GioRegs.EXTINTINTPEND.bit.INTPEND3 = 1;  
// Writing a 1 into a bit will actually clear the interrupt flag!  
// PWM output will turn back on
```