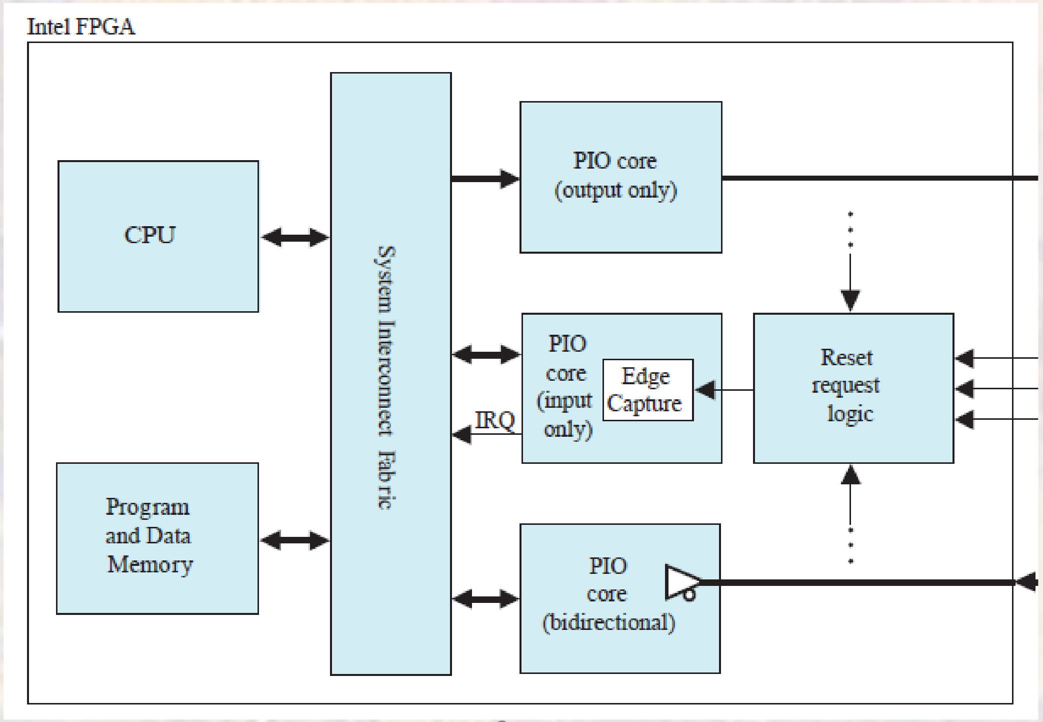
NIOS Peripherals
Parallel I/O

Last updated 7/20/23

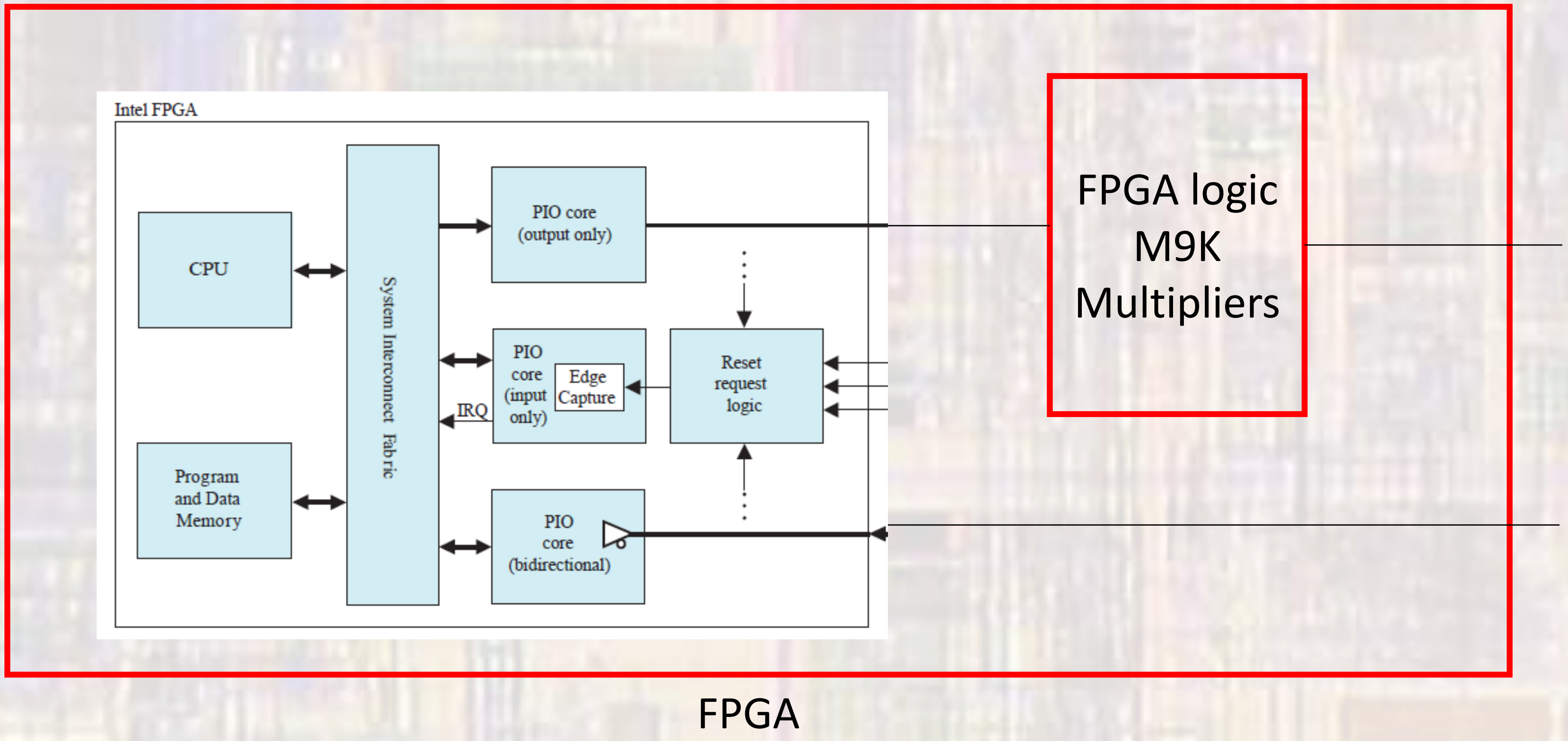
NIOS Peripherals – Parallel I/O - HW

- Parallel IOs (PIO)
  - Each PIO can support up to 32 I/O ports
  - Multiple PIOs can be instantiated
  - Communication to/from the PIOs is via the Altera System Fabric



NIOS Peripherals – Parallel I/O - HW

- Parallel IOs (PIO)
  - Connections can be on chip or off chip



NIOS Peripherals – Parallel I/O - HW

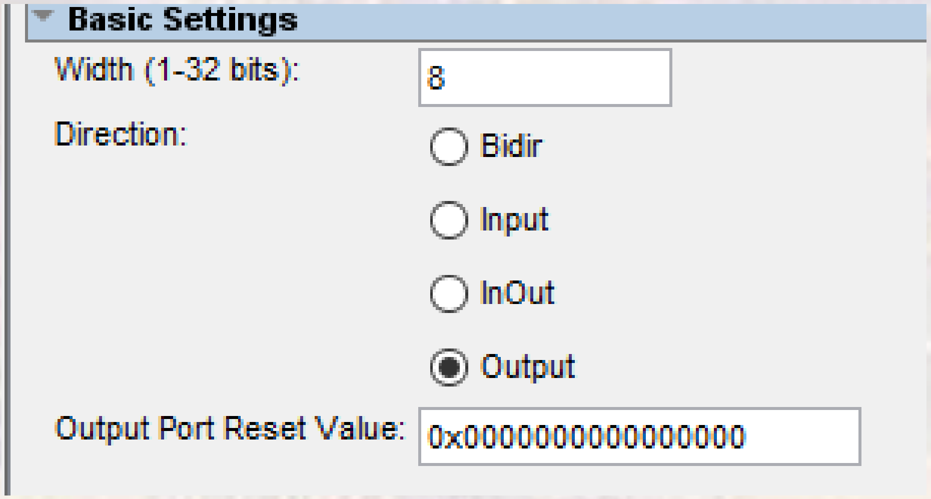
- Parallel IOs (PIO)

  - 4 user visible registers
    - Data
    - Direction
    - Interrupt Mask
    - Edge Capture

  - System interface
    - Avalon Memory Mapped Slave Port
      - Address
      - Data
      - Control
      - Interrupt output

NIOS Peripherals – Parallel I/O - HW

- Parallel IOs (PIO)
  - Basic Settings
  - Width
  - Direction
  - Reset value for outputs



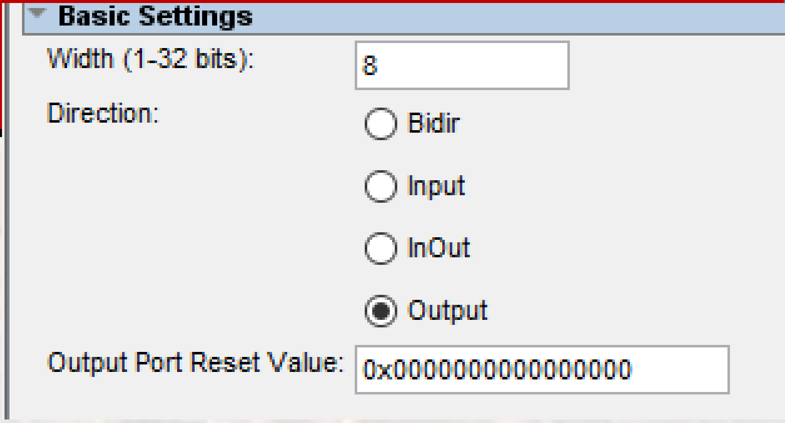
Basic Settings

Width (1-32 bits): 8

Direction: ( ) Bidir ( ) Input ( ) InOut [✓ Output]

Output Port Reset Value: 0x0000000000000000

NIOS Peripherals – Para



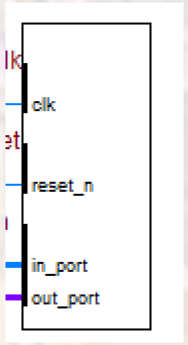
Basic Settings

Width (1-32 bits): 8

Direction: ( ) Bidir ( ) Input ( ) InOut [✓ Output]

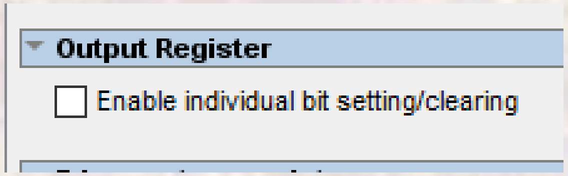
Output Port Reset Value: 0x0000000000000000

- Parallel IOs (PIO)
  - Bidir
    - Single pin for each bit
    - Direction set by the direction register
    - Tri-state by making the pin an input
  - Input
    - Input only
  - InOut
    - Separate busses for input and output
    - Each bus is unidirectional
  - Output
    - Output drive only



NIOS Peripherals – Parallel I/O - HW

- Parallel IOs (PIO)

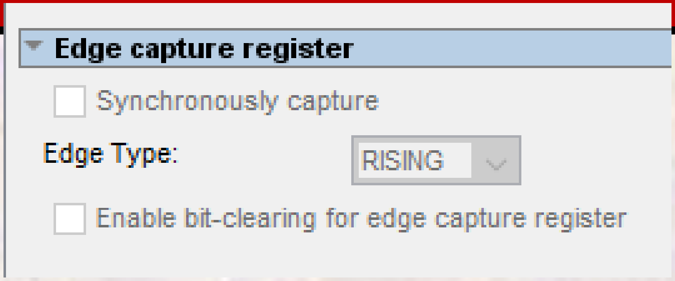


Output Register

☐ Enable individual bit setting/clearing

  - Output Register
    - Enables setting or clearing individual bits
    - Implements 2 additional n bit registers
      - outclear
      - outset

NIOS Peripherals – Parallel I/O - HW



Edge capture register

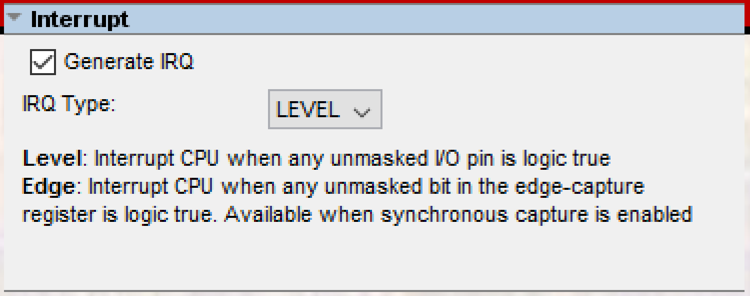
- [ ] Synchronously capture

Edge Type: RISING

- [ ] Enable bit-clearing for edge capture register

- Parallel IOs (PIO)
  - Edge capture
    - Detects and flags a pin event
    - Select Synchronously capture to enable
    - Implements an additional n bit register
      - edgecapture
      - Essentially an interrupt flag register
    - Capture on rising / falling / any edge
    - Enable bit clearing to enable clearing the edgecapture register
      - Write a 1 to the bit you want to clear
      - IF DISABLED – writing to any bit in the register clears all the bits

NIOS Peripherals – Parallel I/O - HW



Interrupt

[✓] Generate IRQ

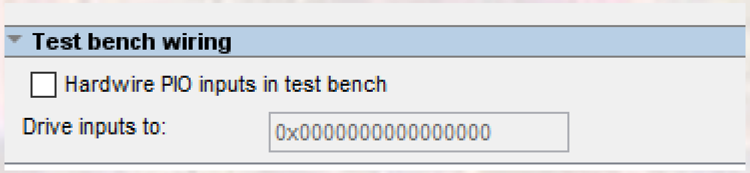
IRQ Type: LEVEL

Level: Interrupt CPU when any unmasked I/O pin is logic true
Edge: Interrupt CPU when any unmasked bit in the edge-capture register is logic true. Available when synchronous capture is enabled

- Parallel IOs (PIO)
  - Interrupt
    - Level
      - creates an interrupt signal when the pin is high
    - Edge
      - creates an interrupt when the edgecapture flag is 1

NIOS Peripherals – Parallel I/O - HW

- Parallel IOs (PIO)
  - Test bench wiring
    - Provide input values for simulation


Test bench wiring

☐ Hardwire PIO inputs in test bench

Drive inputs to: 0x0000000000000000

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - Software file
  - altera_avalon_pio_regs.h
    - defines register map
    - defines constants

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - Register Map

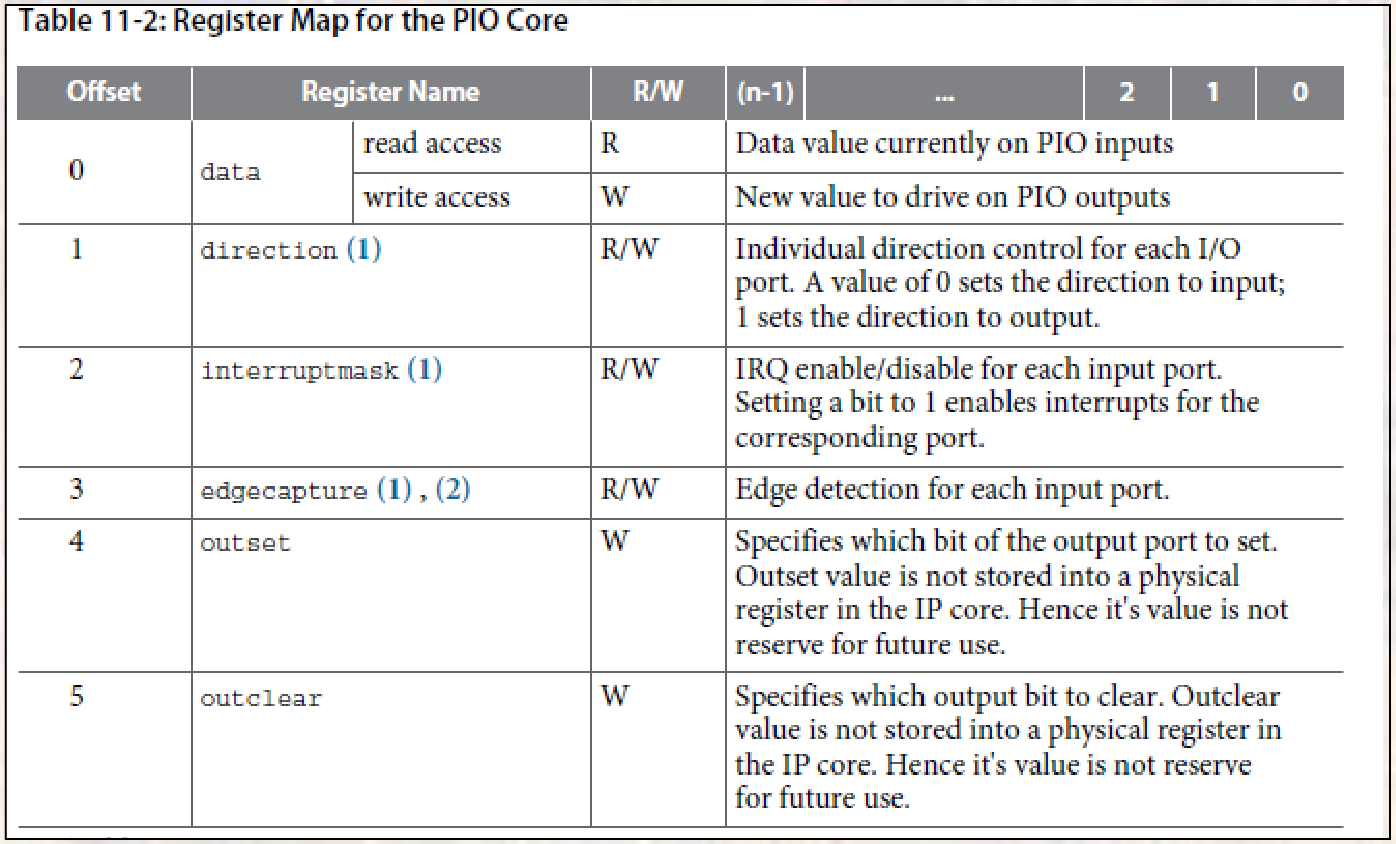


Table 11-2: Register Map for the PIO Core

Offset	Register Name		R/W	(n-1)	...	2	1	0
0	data	read access	R	Data value currently on PIO inputs				
		write access	W	New value to drive on PIO outputs				
1	direction (1)		R/W	Individual direction control for each I/O port. A value of 0 sets the direction to input; 1 sets the direction to output.				
2	interruptmask (1)		R/W	IRQ enable/disable for each input port. Setting a bit to 1 enables interrupts for the corresponding port.				
3	edgecapture (1), (2)		R/W	Edge detection for each input port.				
4	outset		W	Specifies which bit of the output port to set. Outset value is not stored into a physical register in the IP core. Hence it's value is not reserve for future use.				
5	outclear		W	Specifies which output bit to clear. Outclear value is not stored into a physical register in the IP core. Hence it's value is not reserve for future use.				

only exist in certain modes

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - Data Register

  - Read input → provides input value
  - Read output → provides value on output
  - Read bidir → if output – provides value on output
    if input – provides input value
  - Read inout → provides input value

  - Write input → no effect
  - Write output → generates output signal
  - Write bidir → generates output signal for output direction
  - Write inout → generates output signal to the output bus

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - Direction Register

  - sets the direction of the pins in a bidir port
  - 0 → input
  - 1 → output

  - Only exists in bidir mode
    - reading it in any other mode is undefined

  - On reset – all pins are set to input (high Z)

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - Interrupt Mask Register

  - Enables interrupts on individual pins
  - 0 → disabled
  - 1 → enabled

  - Only exists if interrupt is enabled

  - On reset – all pins bits are set to 0 – not enabled

NIOS Peripherals – Parallel I/O - SW

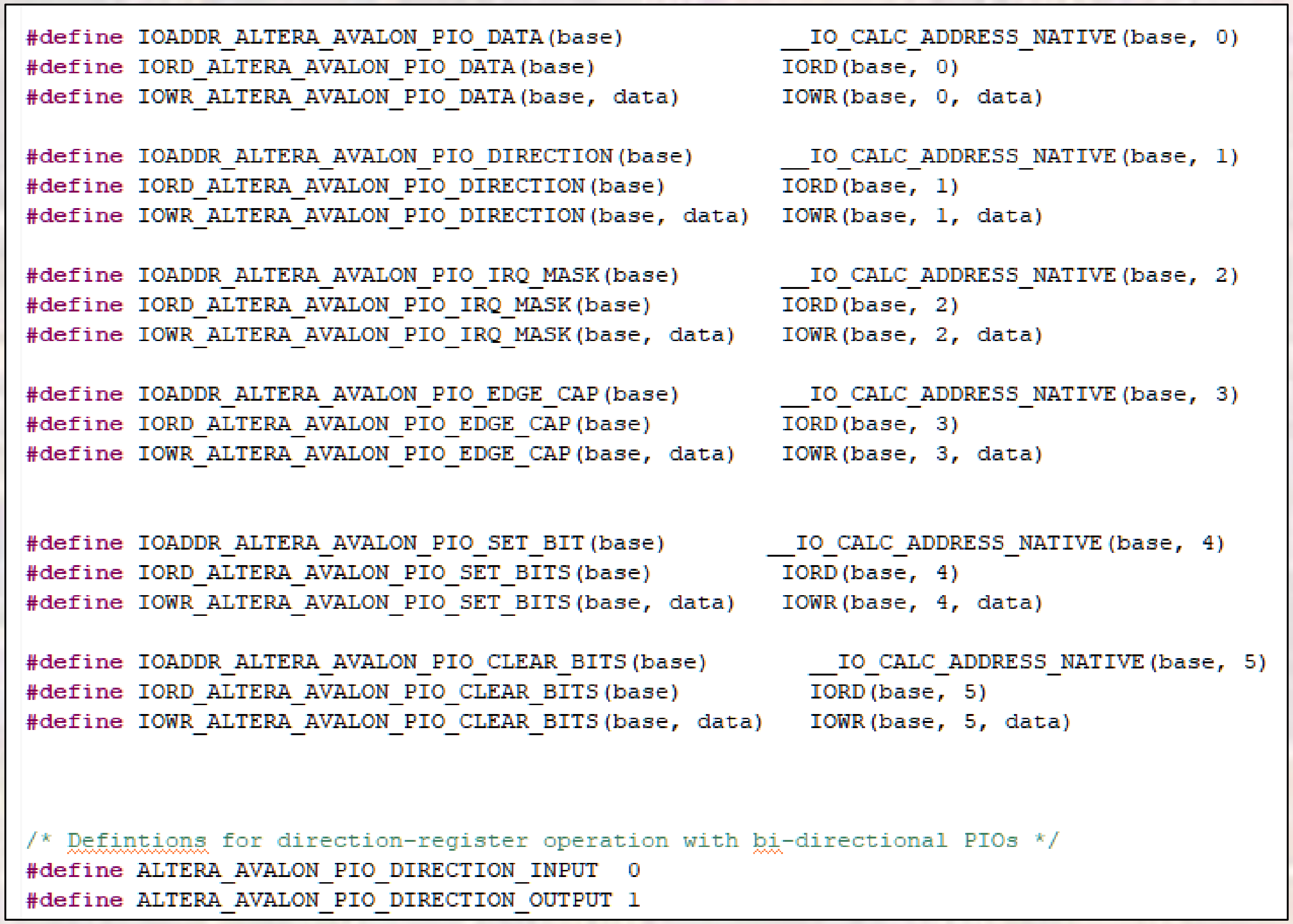
- Parallel IOs (PIO)
  - Edge Capture Register
  - Bit n is set on appropriate event on pin n
  - once set – must be cleared to detect another event
  - enable bit clearing → clear individual bits (flags) via the write to edge capture register
  - no enable bit clearing → all bits (flags) cleared via write to edge capture register
  - Only exists if edge capture is enabled

NIOS Peripherals – Parallel I/O - SW

- Parallel IOs (PIO)
  - outset / outclear Register
  - set bit in outset to set the pin (outputs only)
  - set bit in outclear to clear the pin (outputs only)
  - Not registered – so must maintain signal to maintain state

NIOS Peripherals – Parallel I/O - SW

- altera_avalon_pio_regs.h



```
#define IOADDR_ALTERA_AVALON_PIO_DATA(base)          __IO_CALC_ADDRESS_NATIVE(base, 0)
#define IORD_ALTERA_AVALON_PIO_DATA(base)            IORD(base, 0)
#define IOWR_ALTERA_AVALON_PIO_DATA(base, data)      IOWR(base, 0, data)

#define IOADDR_ALTERA_AVALON_PIO_DIRECTION(base)     __IO_CALC_ADDRESS_NATIVE(base, 1)
#define IORD_ALTERA_AVALON_PIO_DIRECTION(base)       IORD(base, 1)
#define IOWR_ALTERA_AVALON_PIO_DIRECTION(base, data) IOWR(base, 1, data)

#define IOADDR_ALTERA_AVALON_PIO_IRQ_MASK(base)      __IO_CALC_ADDRESS_NATIVE(base, 2)
#define IORD_ALTERA_AVALON_PIO_IRQ_MASK(base)        IORD(base, 2)
#define IOWR_ALTERA_AVALON_PIO_IRQ_MASK(base, data)  IOWR(base, 2, data)

#define IOADDR_ALTERA_AVALON_PIO_EDGE_CAP(base)      __IO_CALC_ADDRESS_NATIVE(base, 3)
#define IORD_ALTERA_AVALON_PIO_EDGE_CAP(base)        IORD(base, 3)
#define IOWR_ALTERA_AVALON_PIO_EDGE_CAP(base, data)  IOWR(base, 3, data)


#define IOADDR_ALTERA_AVALON_PIO_SET_BIT(base)      __IO_CALC_ADDRESS_NATIVE(base, 4)
#define IORD_ALTERA_AVALON_PIO_SET_BITS(base)        IORD(base, 4)
#define IOWR_ALTERA_AVALON_PIO_SET_BITS(base, data)  IOWR(base, 4, data)

#define IOADDR_ALTERA_AVALON_PIO_CLEAR_BITS(base)       __IO_CALC_ADDRESS_NATIVE(base, 5)
#define IORD_ALTERA_AVALON_PIO_CLEAR_BITS(base)         IORD(base, 5)
#define IOWR_ALTERA_AVALON_PIO_CLEAR_BITS(base, data)   IOWR(base, 5, data)



/* Defintions for direction-register operation with bi-directional PIOs */
#define ALTERA_AVALON_PIO_DIRECTION_INPUT  0
#define ALTERA_AVALON_PIO_DIRECTION_OUTPUT 1
```