

GPIO

ATmega1284P

Networks and Embedded Software

Module 4.3.3

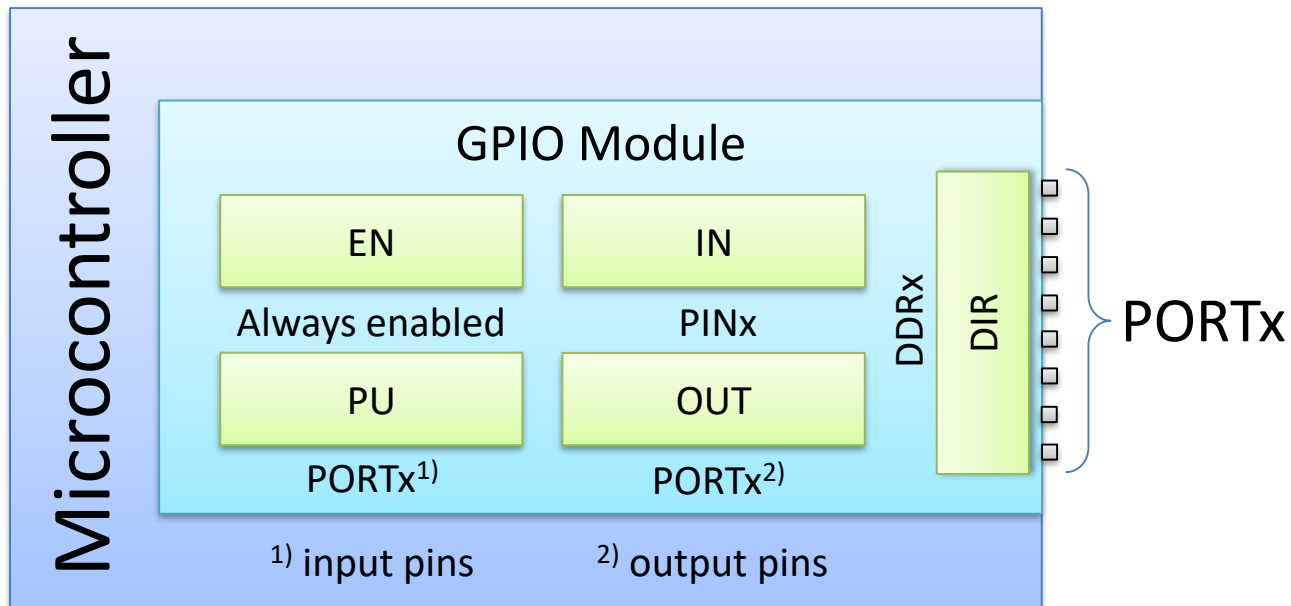
by Wolfgang Neff

GPIO (1)

- Implementation
 - Pins per Port: 8
 - Name of Ports: A, B, C, D
 - Naming Scheme: REGx
 - Enabling: Enabled by default

GPIO (2)

- Register Mapping



¹) input pins

²) output pins

x: A,B,C,D

GPIO (3)

- DDR: Data Direction Register



- Bit 7:0 – DDR: Data Direction
 - 0: pin configured for input
 - 1: pin configured for output
- Naming Scheme
 - **DDR** xn (x: port name, n: bit position)

GPIO (4)

- PORT: Data Output Register



– Bit 7:0 – PORT: Data Output Value

- 0: pin is driven low (outputs 0)
- 1: pin is driven high (outputs 1)

– Naming Scheme

- **PORTxn** (x: port name, n: bit position)

GPIO (5)

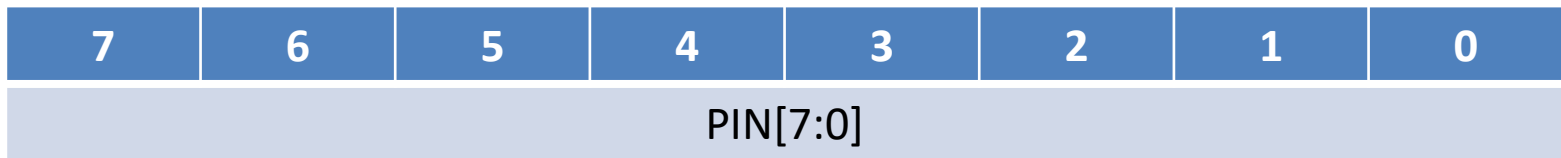
- PORT: Pull-Up Enable Register



- Bit 7:0 – PORT: Pull-Up Enable
 - 0: pull-up resistor is disabled
 - 1: pull-up resistor is enabled
- Register PORT has two functions
 - $DDR[n] = 0$: pull-up resistor control
 - $DDR[n] = 1$: output value

GPIO (6)

- PIN: Data Input Value Register



– Bit 7:0 – PIN: Data Input Value

- 0: pin indicates low voltage (input is 0)
- 1: pin indicates high voltage (input is 1)

– Naming Scheme

- **PIN** x_n (x: port name, n: bit position)

GPIO (7)

- Configuration Example

- Active low LED on pin 0 and 1 of port A

- Initialization

Initialisation is often done by assignment

- Data direction is output

- `DDRA = ((1 << DDRA0) | (1 << DDRA1));`

- Initial state: LED 0 -> on, LED 1 -> off

- `PORTA = (1 << PORTA1);`

GPIO (8)

- Configuration Example (continued)

- Operation

- Switch LED 0 off and LED 1 on

- `PORTA |= (1 << PORTA0);`

- `PORTA &= ~(1 << PORTA1);`

Change is done by bit manipulation

GPIO (9)

- Configuration Example
 - Active low button on pin 0 of port A
 - Initialization
 - Data direction is input
 - `DDRA = 0x00;`
 - Activate pull up resistor
 - `PORTA = (1 << PORTA0);`

GPIO (10)

- Configuration Example (continued)

- Operation

- Read the active low button

- `if (!(PINA & (1 << PINA0))) {...}`



The button is active low. So we have to **test (&)** if the **bit is logically not (!)** set.

GPIO (11)

- Register Summary (continued)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTx	PORTx7	PORTx6	PORTx5	PORTx4	PORTx3	PORTx2	PORTx1	PORTx0
DDRx	DDRx7	DDRx6	DDRx5	DDRx4	DDRx3	DDRx2	DDRx1	DDRx0
PINx	PINx7	PINx6	PINx5	PINx4	PINx3	PINx2	PINx1	PINx0
MCUCR	JDT	BODS	BODSE	PUD	-	-	IVSEL	IVCE
EICRA	-	-	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00
EIMSK	-	-	-	-	-	INT2	INT1	INT0

GPIO (12)

- Interrupt Summary

Source	Description
INT0_vect	External interrupt vector 0
INT1_vect	External interrupt vector 1
INT2_vect	External interrupt vector 2

