

# Ports: Eingabe und Ausgabe mit MBED

## 1. Ausgabe

### a. Parallele Ausgabe:

- Ausgabeport initialisieren:

PortOut *meinPortname*(Port,Maske);

Beispiel:

int main()

{ **//8-Bit Port mit dem Namen ampel an GPIOC**

**PortOut ampel(PortC,0xFF);**

ampel=0b00000001; //PC\_0 =1, PC\_1 .. PC\_7 =0

while (true) { //Endlosschleife

ampel=ampel<<1; //Lauflicht, die 1 wird um eine Stelle nach  
//links geschoben

if (ampel==0) ampel=1; //Bei 0: von vorne

wait\_ms(500); //500ms warten

}

}

Mögliche Werte für Port: PortA (für GPIOA), PortB (für GPIOB) PortC (für GPIOC). Mit *Maske* können die Bits ausgewählt werden, die ausgegeben werden sollen. Beispiel: 0xFF bzw. 0b11111111 bedeutet, dass nur die Bits 0 .. 7 ausgegeben werden.

- Ausgabeport verwenden:

Der Port, im Beispiel *ampel*, kann wie eine beliebige int-Variable verwendet werden.

### b. Einzelne Bits ausgeben:

- Ausgabebit initialisieren:

DigitalOut *meinAnschlussname*(Portbezeichnung);

Mögliche Portbezeichnungen sind:

PA\_0 .. PA\_15, PB\_0 .. PB\_15, PC\_0 .. PC\_15

Für den Anschlussnamen sind beliebige Bezeichnungen möglich.

Sonderzeichen, Leerzeichen und Umlaute sollen aber vermieden werden

Beispiel: DigitalOut roteLED(PC\_0); //Die rote LED ist an GPIOC Bit 0  
angeschlossen.

Bei Bedarf können mit roteLED.mode(*PinMode*); weitere Einstellungen  
vorgenommen werden: PullUp, PullDown, PullNone, OpenDrain

- Ausgabebit verwenden: Beliebige Zuweisungen sind möglich.

Beispiele:     roteLED=1;  
                 roteLED=0;  
                 int x=1;  
                 roteLED=x;  
                 roteLED=grueneLED;  
                 roteLED=!roteLED;  
                 roteLED=true;  
                 roteLED=false;

## 2. Eingabe

### a. Parallele Eingabe

- Eingabeport initialisieren:

```
PortIn meinPortname(Port,Maske);
```

Beispiel:

```
int main()
{
    PortOut ampel(PortC,0xFF);
    PortIn schalterchen(PortB,0b11111111);
    schalterchen.mode(PullDown);
    ampel=1;
    while (true) {
        ampel=schalterchen;
    }
}
```

Mögliche Werte für Port: PortA (für GPIOA), PortB (für GPIOB) PortC (für GPIOC). Mit *Maske* können die Bits ausgewählt werden, die eingegeben werden sollen. Beispiel: 0xFF bzw. 0b11111111 bedeutet, dass nur die Bits 0 .. 7 eingegeben werden. In diesem Fall machen die Schalterchen eine Verbindung nach „1“. Offenen Schalter sollen „0“ sein. Deshalb:

- Eingabeport verwenden:

Der Port, im Beispiel *schalterchen*, kann wie eine beliebige int-Variable verwendet werden.

### b. Einzelne Bits einlesen:

- Eingabebit initialisieren:

```
DigitalIn meinAnschlussname(Portbezeichnung);
```

Mögliche Portbezeichnungen sind:

PA\_0 .. PA\_15, PB\_0 .. PB\_15, PC\_0 .. PC\_15

Für den Anschlussnamen sind beliebige Bezeichnungen möglich.

Sonderzeichen, Leerzeichen und Umlaute sollen aber vermieden werden

Beispiel: DigitalIn Taste(PA\_1); //Die Taste ist an GPIOA Bit 1 angeschlossen.

Bei Bedarf können mit Taste.mode(*PinMode*); weitere Einstellungen vorgenommen werden: PullUp, PullDown, PullNone.

- Eingabebit verwenden: Wie eine Variable.

Beispiel:

```
int main()
{
    PortOut ampel(PortC,0xFF);
    DigitalOut led(PA_5);
    DigitalIn Taste(PA_1);
    Taste.mode(PullDown);
    while (true) {
        led=Taste;
        if (Taste==true) ampel=0b10101010;    //auch möglich: Taste==1
        else ampel=0b01010101;
    }
}
```

3. Aufgaben: Schreiben Sie die Codezeilen in der Programmiersprache CPP
- a. Wenn PA\_1=1 dann an GPIOC die Zahl 35 ausgeben
  - b. Der Ausgabewert an GPIOC wird um 5 erhöht, wenn 25 überschritten wird, wird wieder bei 0 angefangen.
  - c. Nur die unteren 4 Bit von den Eingängen von GBIOB sollen auf GPIOC ausgegeben werden
  - d. Wenn PB\_3=0 ist soll an Bit 0 von GPIOC eine 1 ausgegeben werden
  - e. Setzen Sie die Bits 0,2,4,6 von GPIOC auf 1 und gleichzeitig die Bits 1,3,5,7 auf 0
  - f. Negieren Sie GPIOC bitweise
  - g. Wenn der Ausgang PC\_3=1 ist sollen die Ausgänge von GPIOA den Wert 0 annehmen.
  - h. Wenn PA\_2 und PA\_3 gleichzeitig 1 sind sollen die alle 16 Ausgänge von GPIOC auf 1 gesetzt werden.
  - i. Wenn PA\_1 = 1 wird dann GPIOC um 1 hochzählen. Danach warten bis PA\_1 wieder 0 wird
  - j. Warten auf PA\_10 =1, während gewartet wird soll der Mikrocontroller nichts tun
  - k. Warte auf Taste PA\_10=1, danach
    - Warte auf Taste PA\_6=1, danach
      - 1. Zähle GPIOC um 5 hoch