Informatikwerkstatt, Foliensatz 1a
Wiederholung und Ergänzung

G. Kemnitz

Institut für Informatik, TU Clausthal (IW-F1a)

26. Oktober 2020

Wiederholung

Hexadezimal- und Binärzahlen

1. Zuordnung der Hexadezimalziffern:

bin.	hex.	bin.	hex.	bin.	hex.	bin.	hex.
0000		0100		1000		1100	
0001		0010		1101		0110	
1010		0111		0011		1110	
0101		1111		1011		1001	

2. Umrechnung nach binär:

```
0x15   = 0b . . . | . . .
0xAF2  = 0b . . . | . . . | . . .
0xABCD = 0b . . . | . . . | . . . | . . .
```

3. Umrechnung nach hexadezimal:

```
0b10010110110   = 0x . . .
0b0100111001011 = 0x . . .
0b00110110      = 0x . . .
```

Lösung

Bitverarbeitung

x_1	x_0	$\bar{x}_0$	$x_1 \wedge x_0$	$x_1 \vee x_0$	$x_1 \oplus x_0$
0	0				
0	1				
1	0				
1	1				

```
uint8_t a, b, c, d, e, f, g;
...
a = 0x3E;              //a=0b . . . | . . .
b = a & 0b11100010;    //b=0b . . . | . . . =0x .
c = b | 0b10010001;    //c=0b . . . | . . . =0x .
d = c ^ 0b01100111;    //d=0b . . . | . . . =0x .
e = ~d;                //e=0b . . . | . . . =0x .
f = e >> 2;            //f=0b . . . | . . . =0x .
g = f << 1;            //g=0b . . . | . . . =0x .
```

Vervollständigen Sie die Kommentare.

Lösung

Aufgaben

Aufgabe: Matrix-Tastatur (Experten)

Stecken Sie das dargestellte Tastenfeld PmodKYPD an einen freien Port des Mikrorechnerboards und schreiben Sie ein Programm, dass bei Tastenbetätigung den Hex-Wert der Taste auf die LEDs an Port J ausgibt.

Hinweise:

- Dokumentation und Schaltplan PmodKYPD: [Manual], [Schematic].
- Da es sich um eine Schaltermatrix handelt, erfordert die Abfrage einen Automaten.

Programm mit &, |, ^, >>, <<

```
#include <avr/io.h>
int main(void){
  DDRA =          ;   // Port A (Schalter) Eingänge
  DDRJ =          ;   // Port J (LEDs) Ausgänge
  uint8_t a, b, c, d, e;   // 8-Bit-Variablen
  while(...){             // Endlosschleife
    a =                 ; // Schalterwerte
    b =                 ; // b.Bit0 <= SW1
    c =                 ; // c.Bit0 <= SW2
    d =                 ; // d.Bit0 <= SW3
    e =                 ; // e.Bit0 <= SW4
// Ausgabe: LD0 = (SW1 und SW2) oder (SW3 und SW4)
    PORTJ =
  }
}
```

Programm mit if, else

```
#include <avr/io.h>
int main(void){
  DDRA =          ;   // Port A (Schalter) Eingänge
  DDRJ =          ;   // Port J (LEDs) Ausgänge
  uint8_t a       ;   // 8-Bit-Variablen
  while(...){         // Endlosschleife
    a =             ; // Schalterwerte
    if (              )
                      // PORTJ.Bit0 setzen
    else
                      // PORTJ.Bit0 löschen
  }
}
```

Programm mit switch und case

sw	LD0	sw	LD0	sw	LD0	sw	LD0
0000		0100		1000		1100	
0001		0101		1001		1101	
0010		0110		1010		1110	
0011		0111		1011		1111	

```
...
while(...){                    // Endlosschleife
  a =                     ;    // Schalter lesen
  switch (                     )
    case                   :   // 1. Fall Ausgabe 0
    case                   :   // 2. Fall Ausgabe 0
    case                   :   // 3. Fall Ausgabe 0
    case                   :   // 4. Fall Ausgabe 0
                               // PORTJ.Bit0 setzen
    break;
    default:
                               // PORTJ.Bit0 setzen
   }
}
```