# Ein Bild, das Text, Screenshot, Rechteck, Diagramm enthält. Automatisch generierte BeschreibungBitmanipulation & Tasterabfrage

Programmieren lernen mit dem nanoBoard – AB 13

## Einführung digitaler Verknüpfungen und deren Wahrheitstabellen

Die Kenntnis digitaler Verknüpfungen ist wesentliche Grundlage für die Arbeit mit digitalen Schaltungen oder Mikrocontrollern.

Klären Sie für sich die nachfolgenden Fragen zu den Verknüpfungen:

AND, OR, Nicht (nur ein Eingang), NAND, NOR, EXOR (Antivalenz) und Äquivalenz, die jeweils zwei Eingänge und einen Ausgang haben sollen:

* Was bedeuten diese digitalen Verknüpfungen inhaltlich?
* Was ist eine Wahrheitstabelle und wie sehen diese für die obigen Verknüpfungen aus?
* Wie sind die Symbole dieser digitalen Verknüpfungen für Schaltpläne?

## Hinweis

Für die Bearbeitung der nachfolgenden Aufgaben muss Ihnen das entsprechende Blatt der Formalsammlung vorliegen.

Bei allen Bytes, die vom Entwicklungssystem zur Verfügung gestellt werden, haben die einzelnen Bits eine „eigene Bezeichnung“. Für die einzelnen Bits des Ports C gilt:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PORTC = | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |

Für das Byte ADMUX, das bei der Messung analoger Spannungen eine wesentliche Rolle spielt, gilt:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ADMUX = | REFS1 | REFS0 | ADLAR | MUX4 | MUX3 | MUX2 | MUX1 | MUX0 |

Diese „Namen“ der einzelnen Bits können zu deren Beeinflussung (Manipulation) genutzt werden. Gemäß Formelsammlung können die Bits *gesetzt*, *gelöscht* oder *invertiert* werden.

Programmieren lernen mit dem nanoBoard – AB 13

## Ein Bild, das Text, Screenshot, Rechteck, Diagramm enthält. Automatisch generierte BeschreibungAufgabe Bitmanipulation.1

Wozu führt die Zeile: PORTC |= ((1 << PC5) | (1 << PC3) | (1 << PC1)); ?

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Zustand an PortC(PORTC=13;) |  |  |  |  |  |  |  |  |
| Veränderung … ??? |  |  |  |  |  |  |  |  |
| bitweises **ODER** |  |  |  |  |  |  |  |  |

Wozu führt die Zeile: PORTC &= ~((1 << PC5) | (1 << PC3) | (1 << PC1)); ?

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Zustand an PortC(PORTC=0x63;) |  |  |  |  |  |  |  |  |
| Veränderung … ??? |  |  |  |  |  |  |  |  |
| bitweises **UND** |  |  |  |  |  |  |  |  |

Wozu führt die Zeile: PORTC ^= ((1 << PC6) | (1 << PC2) | (1 << PC0)); ?

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Zustand an PortC(PORTC=45;) |  |  |  |  |  |  |  |  |
| Veränderung … ??? |  |  |  |  |  |  |  |  |
| bitweises **XOR** (exklusiv ODER) |  |  |  |  |  |  |  |  |

## Aufgabe Bitmanipulation.2

Notieren Sie **die jeweils notwendige Zeile**, die PC1 und PC4 setzt, löscht bzw. invertiert.

**Beispiel:**

PB1 und PB4 setzen 🡪 **PORTB |= ((1<<PB1) | (1<<PB4));**

## Aufgabe Bitmanipulation.3

Notieren Sie zum Byte PROL die Zeilen, die die folgende Aufgabe erfüllen:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PROL = | OP | EL | SE | AT | AU | DI | MAZ | DA |

1. Setzen der Bits EL, AU und SE.
2. Invertieren/Toggeln der Bits OP, DI und MAZ
3. Löschen der Bits AT, DI, MAZ und DA
4. Zuweisen des Werts 167 als hexadezimale bzw. binäre Zahl.

Programmieren lernen mit dem nanoBoard – AB 13

## Aufgabe Bitmanipulation.4

Sie müssen auf das allgemein bekannte Byte FIFI Einfluss nehmen.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| FIFI = | KNO | CH | EN | TOL3 | TOL2 | TOL1 | TOL0 | AAA |

Es gilt: **FIFI = 138**

Notieren Sie FIFI als binäre Zahl und notieren Sie, wie sich die einzelnen Zeichen auf FIFI auswirken. Notieren Sie die jeweils entstehenden binären und dezimalen Zahlen. Die Zeilen sollen unabhängig voneinander betrachtet werden.

1. FIFI &= ~((1 << CH) | (1 << TOL1) | (1 << TOL0));
2. FIFI = FIFI ^ ((1 << KNO) | (1 << EN) | (1 << TOL1) | (1 << AAA));
3. FIFI = 0b11001011;
4. FIFI |= ((1 << EN) | (1 << TOL2) | (1 << TOL0) | (1 << AAA));

## Aufgabe Bitmanipulation.5

Programmieraufgabe:

Lassen Sie nacheinander verschiedene Bitmanipulationen ausführen, die als Veränderung an den LEDs (PORT B) zu erkennen sind. Die Veränderungen sollen auf dem seriellen Monitor „angekündigt“ werden. Das Programm soll anschließend von vorne beginnen.

## Erweiterung für „erfahrene“ Programmierer

Durch die vier Taster auf dem µC-Board (S1 bis S4) sollen Veränderungen an den LEDs auf dem µC-Board durchgeführt werden. Alle Beeinflussungen an den LEDs sollen mit Hilfe der Bitmanipulationen erfolgen (Setzen, Löschen, Invertieren). Die Manipulationen sollen auf dem seriellen Monitor angekündigt werden.

Ein Bild, das Text, Screenshot, Rechteck, Diagramm enthält.

Automatisch generierte Beschreibung