

Arbeitsblatt Theorie

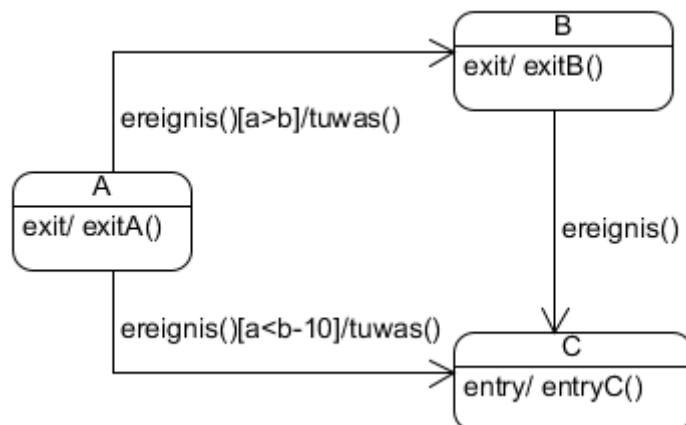
- Die Bewegung eines Roboterfahrzeugs erfolgt gemäß folgender Ausgabe an PortC:

	PortC
Initialisierung	0b00000
Stopp	0b10000
Linksdrehen	0b10110
Rechtsdrehen	0b11001
Vorwärts	0b11010
Rueckwärts	0b10101

Programmablauf:

Nach dem Programmstart steht der Roboter. Die Bewegung beginnt mit dem Ereignis `start()` (Tastendruck): 5 Sekunden vorwärts, 1 Sekunde Linksdrehen, 5 Sekunden rückwärts, 1 Sekunde Rechtsdrehen, dann wieder stopp. Die Zustandsänderungen werden von den Ereignissen `start()` bzw. `stopp()` (Tasten-Interrupts) und vom `zeitereignis()` Timerinterrupt ausgelöst. Das Ereignis `stopp()` beendet jede Bewegung. Nach Stopp beginnt die Bewegung mit Start von vorne.

- Zeichnen Sie das Zustandsdiagramm, der
 - Definieren Sie die Zustände in der Programmiersprache C
 - Deklarieren Sie die Zustandsvariable `zustand` als Ausgangsport
 - Schreiben Sie den C-Code für `start()`
 - Schreiben Sie den C-Code für `stopp()`
 - Schreiben Sie den C-Code für `zeitereignis()`
- Wie stellen Sie sicher, dass ein Ereignis nur in einem bestimmten Zustand und bei einer bestimmten Bedingung eine Reaktion auslöst.
 - Wie werden Ereignisse im Zustandsdiagramm dargestellt?
 - Wie unterscheiden sich interne Ereignisse von Selbsttransitionen
 - Schreiben Sie die Operation `ereignis()` in der Programmiersprache C. Verwenden Sie die Zustandsvariable `int zustand`.



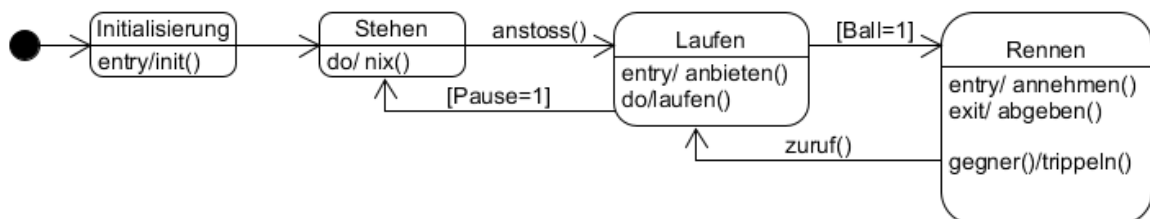
- Welchen Vorteil hat es, wenn für die Zustandsvariable direkt ein Ausgangsport des Mikrocontrollers verwendet werden kann.
- Wie wird ein Schaltwerk ohne Ausgangsschaltnetz analog auf dem Mikrocontroller abgebildet?

8. Rekonstruieren Sie das Zustandsdiagramm:

```
int main(void)
{
    zustand=Initialisierung;
    init();
    zustand=A;
    while(true)
    {
        switch(zustand)
        {
            case A: tuwas();
                    if (taste==1)
                    {
                        op1();
                        zustand=B;
                        op2();
                    }
                    break;
            case B: if (taste==0)
                    {
                        zustand = C;
                        op3();
                    }
                    break;
            case C: op4();
                    break;
        }
    }
}
```

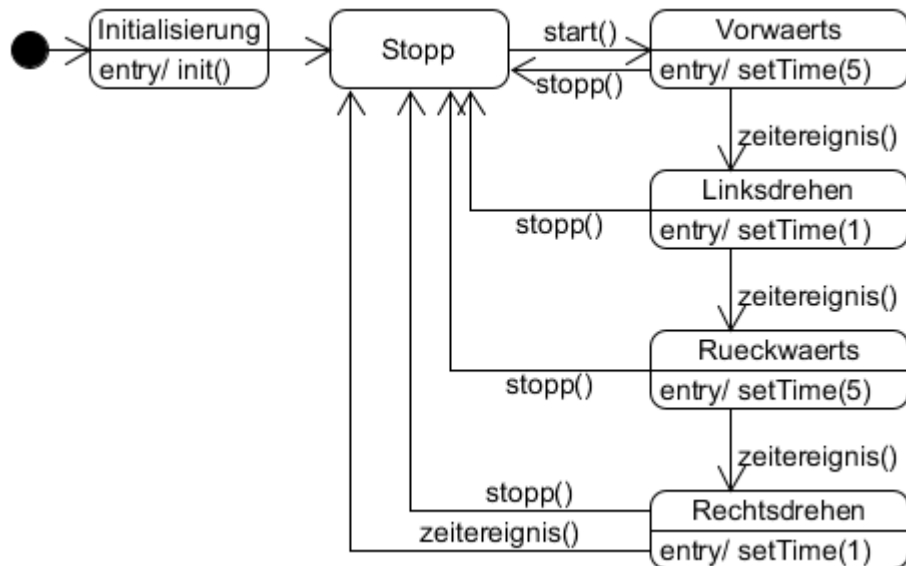
9. Analysieren Sie folgendes Zustandsdiagramm:

- Schreiben Sie das Hauptprogramm int main(void)
- Wobei handelt es sich um ein internes Ereignis?
- Definieren Sie die Zustände
- Schreiben Sie die Ereignisprogramme (ISR) von anstoss() und zuruf()
- Schreiben Sie das Ereignisprogramm (ISR) für gegner()



Lösungsvorschlag:

1.



```
#define Initialisierung 0b000000
#define Stopp 0b10000
#define Linksdrehen 0b10110
#define Rechtsdrehen 0b11001
#define Vorwaerts 0b11010
#define Rueckwaerts 0b10101
PortOut zustand(PortC,0b11111);
InterruptIn TasteStart(PA_1);
InterruptIn TasteStopp(PA_6);
```

```
void setTime(int pT)
{ TIM6->ARR=pT*1000;
  TIM6->CNT=0;
}
```

```
void zeitereignis(void)
{ switch(zustand)
{
    case Vorwaerts:
        zustand=Linksdrehen;
        setTime(1);
        break;
    case Linksdrehen:
        zustand=Rueckwaerts;
        setTime(5);
        break;
    case Rueckwaerts:
        zustand=Rechtsdrehen;
        setTime(1);
        break;
```

```

        case Rechtsdrehen:
            zustand=Stopp;
            break;
    }
    TIM6->SR=0;
    HAL_NVIC_ClearPendingIRQ(TIM6_IRQn);
}

void start()
{
    switch(zustand)
    {
        case Stopp:
            zustand=Vorwaerts;
            setTime(5);
            break;
    }
}

void stopp()
{
    switch(zustand)
    {
        case Vorwaerts:
        case Rueckwaerts:
        case Linksdrehen:
        case Rechtsdrehen:
            zustand=Stopp;
            break;
    }
}

void init(void)
{
    TasteStart.mode(PullDown);
    TasteStopp.mode(PullDown);
    TasteStart.rise(&start);
    TasteStopp.rise(&stopp);
    RCC->APB1ENR|=0b10000; //Clock Enable
    TIM6->PSC=31999;      //Prescaler 1ms
    TIM6->ARR=4999;       //Autoreload 5000*1ms = 5s
    TIM6->DIER=1;         //UIE = 1 (Update Interrupt Enable)
    TIM6->SR=0;           //UIF =0 (Update Interrupt Flag)
    TIM6->CR1=1;          //CEN=1 (Counter Enable)
    /* TIM6_IRQn interrupt configuration */
    NVIC_SetVector(TIM6_IRQn, (uint32_t)&zeitereignis);
    HAL_NVIC_EnableIRQ(TIM6_IRQn);
}

```

```

int main()
{
    zustand=Initialisierung;
    init();
    zustand=Stopp;

    while (true) {

    }
}

```

2. Wie stellen Sie sicher, dass ein Ereignis nur in einem bestimmten Zustand und bei einer bestimmten Bedingung eine Reaktion auslöst.

Antwort: Das Ereignisprogramm beginnt mit switch (zustand)

3. Externe Ereignisse werden auf dem Transitionspfeil eingetragen
Interne Ereignisse stehen im Zustandsrechteck am Schluss
4. Bei Selbsttransitionen wird der Zustand verlassen und erneut betreten. Demzufolge werden die exit- und entry-Aktivitäten des Zustands ausgeführt. Beim internen Ereignis verbleibt der Mikrocontroller im Zustand. Entry- und exit-Aktivitäten werden nicht ausgeführt.

5. void ereignis()


```

      {
      switch (zustand)
      {
          case A: if (a>b)
              {
                  exitA();
                  tuwas();
                  zustand=B;
              }
          else if (a<b-10)
              {
                  exitA();
                  tuwas()
                  zustand=C;
                  entryC()
              }
          break;
          case B: exitB();
                  zustand=C;
                  entryC();
                  break;
      }
      }
      
```

6. Welchen Vorteil hat es, wenn für die Zustandsvariable direkt ein Ausgangsport des Mikrocontrollers verwendet werden kann.
Bei einem Zustandswechsel wird der Ausgangsport automatisch auf die passende Bitkombination des betreffenden Zustands eingestellt. Es bedarf keiner zusätzlichen Ausgabe.

7. Wie wird ein Schaltwerk ohne Ausgangsschaltnetz analog auf dem Mikrocontroller abgebildet?
Indem für die Zustandsvariable der Ausgangsport genommen wird und die Zustände entsprechend der gewünschten Bitkombination codiert definiert werden.

Beispiel:

PortOut zustand(PortC,0xFF);

#define A 0b00000000

#define B 0b10101010

Zustandswechsel:

Zustand=B; //0b10101010 wird auf PortC ausgegeben

8. Lösung:



9. Lösung:

a. Hauptprogramm

int main()

```

{
    zustand=Initialisierung;
    init();
    zustand=Stehen;
    while(true)
    {
        switch(zustand)
        {
            case Stehen: nix();
                        break;
            case Laufen: laufen();
                        if (Ball==1)
                        {
                            zustand=Rennen;
                            annehmen();
                        }
                        if (Pause==1)
                        {
                            zustand=Stehen;
                        }
                        break;
            case Rennen: break;
        }
    }
}
  
```

b. Gegner()

c. Zustandsdefinitionen

```
#define Initialisierung 0
#define Stehen          1
#define Laufen          2
#define Rennen          3
```

d. Lösung:

```
void anstoss()
{
    switch (zustand)
    {
        case Stehen:
            zustand=Laufen;
            anbieten();
            break;
    }
}

void zuruf()
{
    switch (zustand)
    {
        case Rennen: abgeben();
                    zustand=Laufen;
                    anbieten();
    }
}
```

e. Lösung:

```
void gegner()
{
    switch (zustand)
    {
        case Rennen: trippeln(); break;
    }
}
```