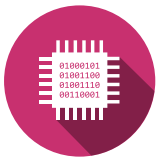


Programzähler

Labor Digitales Design

Inhalt

1 Ziel

Dieses Labor zeigt die Erstellung eines Programm-Codes mit Hilfe eines Festwertspeichers (Read-Only Memory (ROM)) mit Hilfe der Erstellung eines Programmzählers Program Counter (PC).

Es wird auch das Zeichnen von hierarchischen Schaltkreisen geübt.

2 ROM zur Steuerung der ALU

2.1 Schaltung

Die Abbildung Abbildung 1 zeigt eine vereinfachte Darstellung eines Prozessors, mit einer Arithmetic and Logical Unit (ALU), Register und einem Programmzähler.

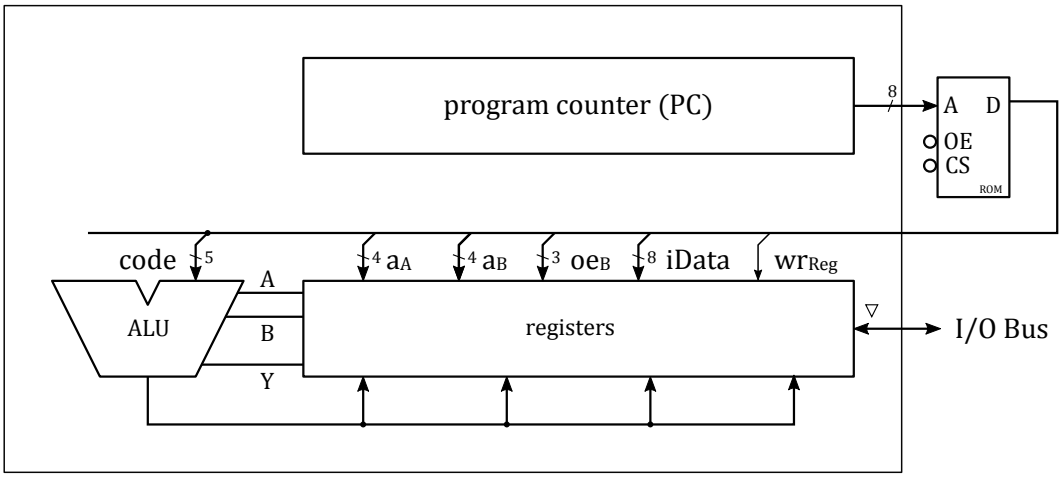


Abbildung 1 - ROM zur Steuerung der ALU und den Registern.abbr

Die Adresse der ROM werden von einem Programmzähler erstellt, welcher es erlaubt, den ROM Inhalt sequenziell durchzulesen. Die Daten der ROM codieren die Steuersignale der ALU und der Register.

Der Eingang Output Enable (OE) steuert den hochohmigen Ausgang der ROM. Der Eingang Chip Select (CS) ist der Selektierungssignal der ROM. Beide müssen aktiv sein, damit der Baustein seine Daten an den Ausgang stellt.

2.2 Operationssequenz

Die Befehle werden in 2 Phasen durchgeführt, wie in Abbildung Abbildung 2 dargestellt.

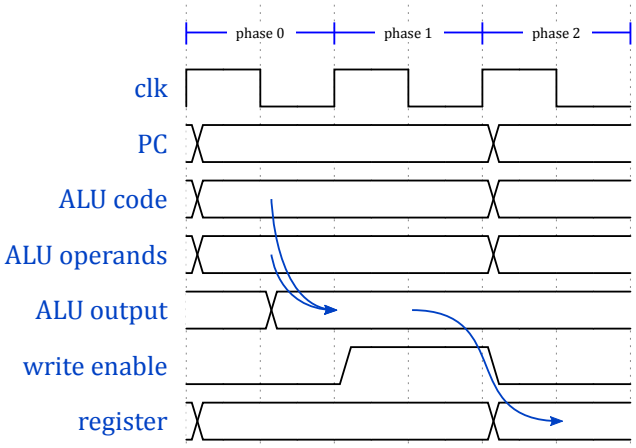
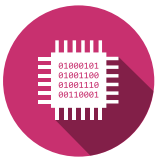


Abbildung 2 - Befehlsphasen

Der ALU Ausgang ist am Ende der Phase 0 stabil. Dieser Wert wird im selektiertem Register bei der steigenden Flanke am Ende der Phase 1 gespeichert.



3 Software-Erstellung eines seriellen Ports

3.1 Serielle Übermittlung

Die Abbildung Abbildung 3 gibt das zeitliche Verhalten der seriellen Übermittlung eines Datenwortes.

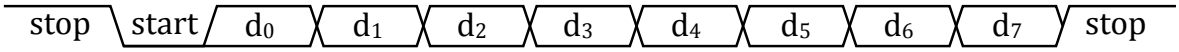


Abbildung 3 - Serielle Übermittlung

Die seriellen Daten werden auf dem niederwertigen Bit des Prozessor-Datenbusses übermittelt.

3.2 Linearer Algorithmus

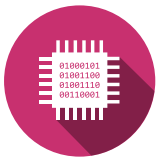
Das Algorithmus, welches im ROM gespeichert ist, ist das folgende:

```
LOAD        s3, FF              ; load stop bit
OUTPUT      s3                  ; output stop bit
LOAD        s3, s3              ; no operation
LOAD        s3, s3              ; no operation
LOAD        s3, s3              ; no operation
LOAD        s3, s3              ; no operation
LOAD        s0, 00              ; load start bit
OUTPUT      s0                  ; output start bit
INPUT       s1                  ; load word to send
OUTPUT      s1                  ; output word, LSB is considered
SR0         s1                  ; shift word, bit 1 -> LSB
OUTPUT      s1                  ; output bit 1
SR0         s1                  ; bit 2 -> LSB
OUTPUT      s1                  ; output bit 2
SR0         s1                  ; bit 3 -> LSB
OUTPUT      s1                  ; output bit 3
SR0         s1                  ; bit 4 -> LSB
OUTPUT      s1                  ; output bit 4
SR0         s1                  ; bit 5 -> LSB
OUTPUT      s1                  ; output bit 5
SR0         s1                  ; bit 6 -> LSB
OUTPUT      s1                  ; output bit 6
SR0         s1                  ; bit 7 -> LSB
OUTPUT      s1                  ; output bit 7
LOAD        s3, s3              ; no operation
OUTPUT      s3                  ; output stop bit
```

3.2.1 Erstellung

Um den linearen Algorithmus zu erstellen ist der PC ein unidirektionaler Zähler. Er kann keinen neuen Wert laden.

Zeichnen Sie die hierarchische Schaltung des Programmzählers des Mikroprozessors. Dieser Zähler muss bei der steigenden Flanke des Taktes inkrementiert werden, wenn **`incPC = '1'`** ist. Beachten Sie, dass in unserem System **`incPC`** jede zweite Taktperiode gesetzt wird. Ignorieren Sie die Steuerung zum Laden eines neuen Wertes im Zähler.



Simulieren Sie das System und verifizieren Sie die Funktionalität des Zählers und des Prozessorsystems.

3.3 Algorithmus mit Schlaufen

Das folgende Algorithmus erlaubt es, einen kompakteren Code zu schreiben:

```
LOAD        s3, FF              ; load stop bit
OUTPUT      s3                  ; output stop bit
LOAD        s2, 04              ; initialize loop counter 3
SUB         s2, 01              ; decrement loop counter 4
JUMP NZ     03                  ; loop back if not end of count 5
LOAD        s0, 00              ; load start bit 6
OUTPUT      s0                  ; output start bit 7
LOAD        s2, 08              ; initialize loop counter 8
INPUT       s1                  ; load word to send 9
LOAD        s3, s3              ; no operation
OUTPUT      s1                  ; output word, LSB is considered
SR0         s1                  ; next bit -> LSB
SUB         s2, 01              ; decrement loop counter
JUMP NZ     0A                  ; loop back if not end of count
OUTPUT      s3                  ; output stop bit
```

3.3.1 Erstellung

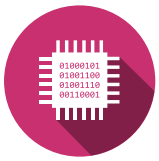
Um den Algorithmus mit Schlaufen durchzuführen soll der PC neue Werte laden können.

Ändern Sie die hierarchische Schaltung des Programmzählers, um das Laden eines neuen Wertes zu ermöglichen.

Simulieren Sie das System und verifizieren Sie die Funktionalität des Zählers und des Prozessorsystems.

3.4 Vergleich

Vergleichen Sie die Übertragungs-Baudrate der beiden Algorithmen.



Glossar

ALU – **Arithmetic and Logical Unit** 2

CS – **Chip Select** 2

OE – **Output Enable** 2

PC – **Program Counter** 1, 3, 4

ROM – **Read-Only Memory** 1, 2, 3