



h_da

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbi
FACHBEREICH INFORMATIK

RECHNERARCHITEKTUR
SS2022
Termin 2
Umgang Befehlssatz eines MU1 Prozessors

| Name, Vorname | Matrikelnummer | Anmerkungen |
|---------------|--------------------|--------------|
| | | |
| | | |
| Datum | Raster (z.B. Mi3x) | Testat/Datum |
| | | |

Legende: V:Vorbereitung, D: Durchführung, P: Protokoll/Dokumentation, T: Testat

Vorbereitung

Bereiten Sie die Lösungen daheim oder in den offenen Laboren so vor, dass Sie die Ergebnisse zum Labortermin präsentieren können.

Aufgabe1:

Erweitern sie den Befehlssatz des MU1 Prozessors um die Befehle PUSH, POP, LDR S, STR S, MOV PC und MOV SP . Zeichnen Sie in die Diagramme den jeweiligen Datenfluss und füllen Sie die Steuerungstabelle aus.

Der Befehl PUSH dekrementiert ($SP=SP-1$) den Stackpointer (Register SP) und speichert den aktuellen Akkumulatorinhalt (Register A) auf dem Stack.

Der Befehl POP lädt den Wert auf den der Stackpointer zeigt in den Akkumulator und inkrementiert ($SP=SP+1$) den Stackpointer.

Der Befehl STR S schreibt den Inhalt des Akkumulator in die Speicherstelle mit der Adresse, welche in der Speicherstelle mit der Adresse S steht.

Der Befehl LDR S lädt den Inhalt der Speicherstelle mit der Adresse, welche in der Speicherstelle mit der Adresse S steht, in den Akkumulator.

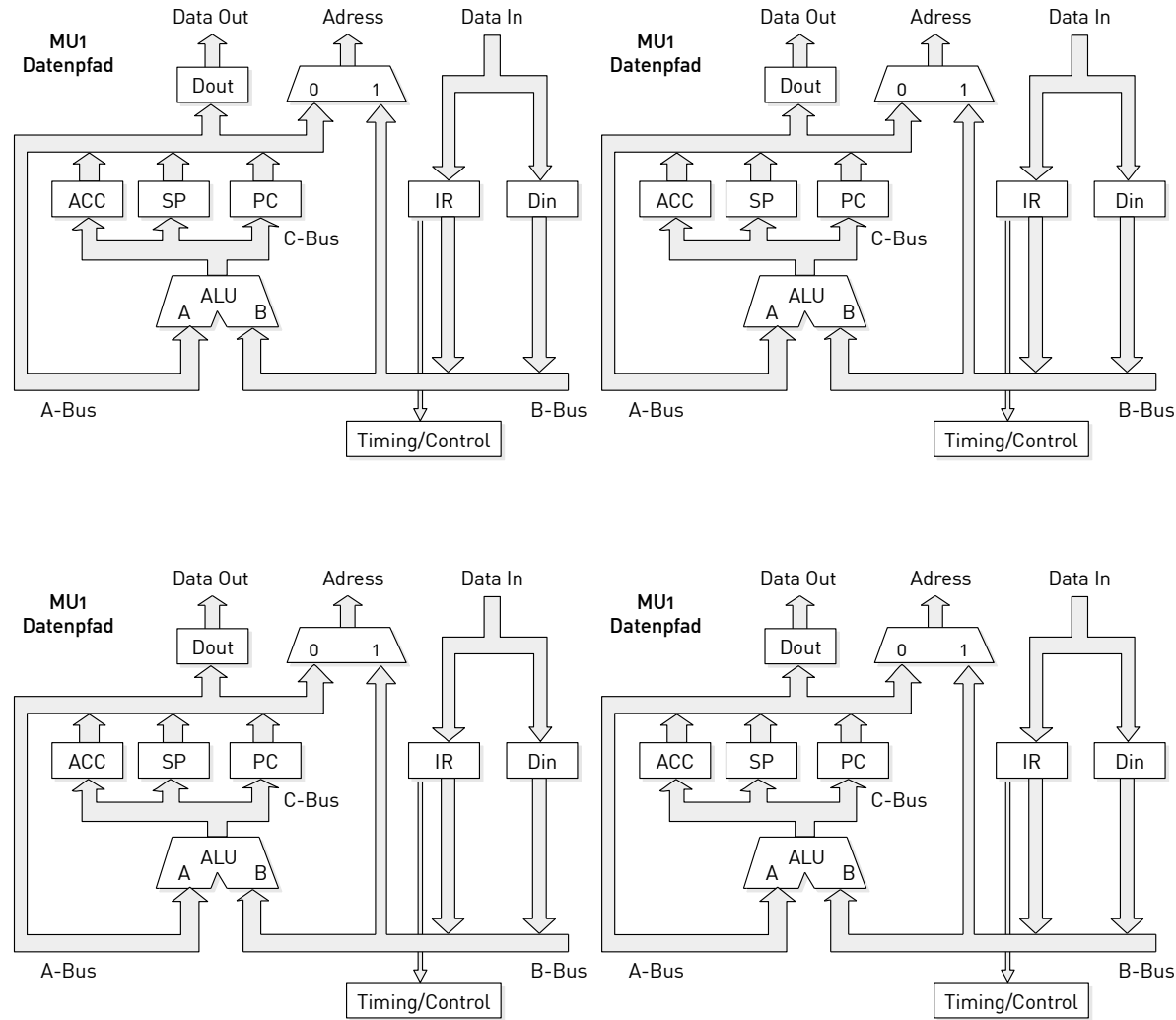
Der Befehl MOV PC kopiert den Inhalt vom Register ACC in das Register PC.

Der Befehl MOV SP kopiert den Inhalt vom Register ACC in das Register SP.

Befehlstabelle für MU1

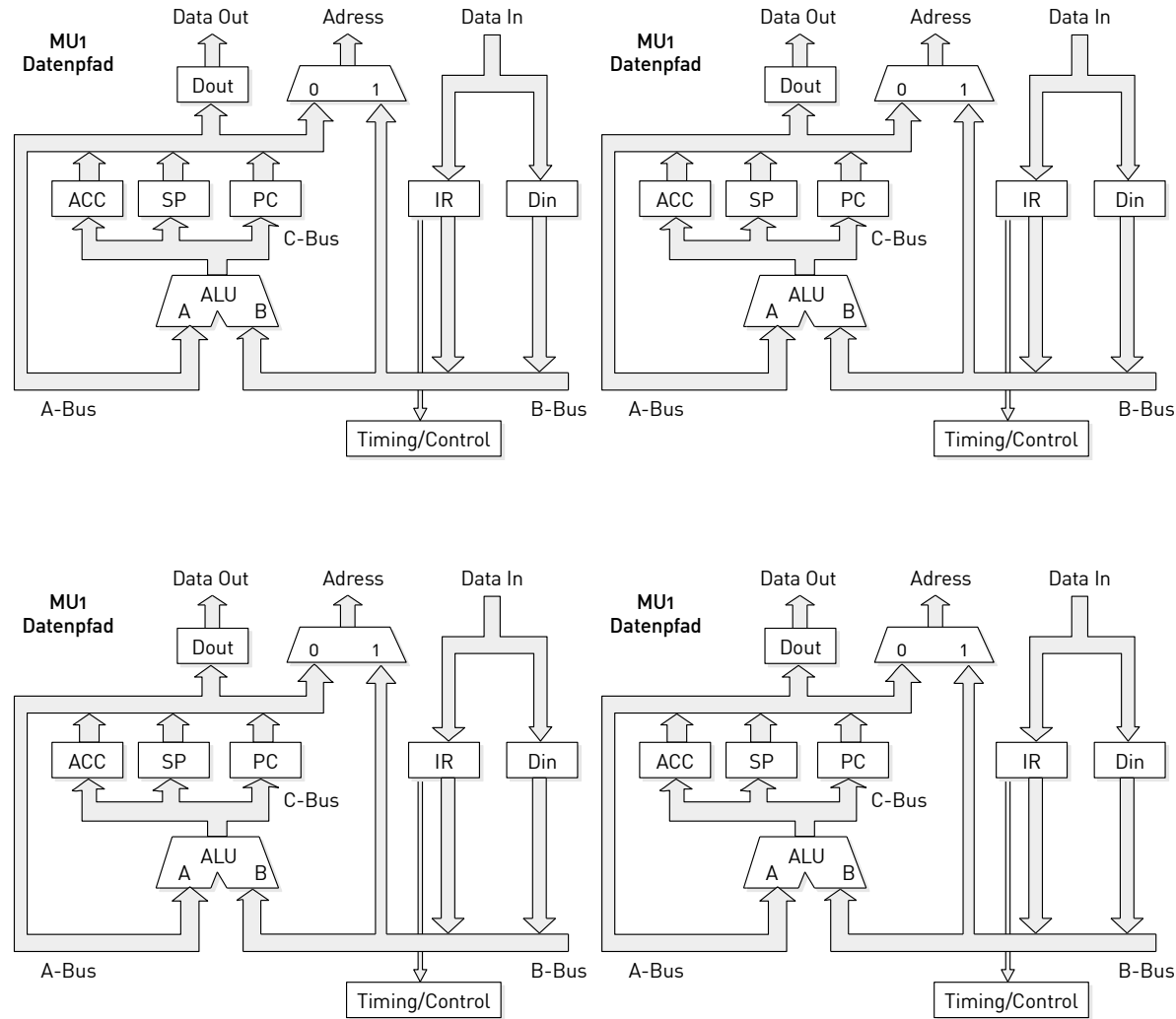
| <i>Instruction</i> | <i>Opcode Bit 15..12</i> | <i>Effekt</i> |
|--------------------|------------------------------|--------------------------------|
| <i>Reset</i> | | $PC = 0$ |
| LDA S | 0000 | $ACC = [S]$ |
| STO S | 0001 | $[S] = ACC$ |
| ADD S | 0010 | $ACC = ACC + [S]$ |
| SUB S | 0011 | $ACC = ACC - [S]$ |
| JUMP S | 0100 | $PC = S$ |
| JGE S | 0101 | IF $ACC \geq 0$ $PC = S$ |
| JNE S | 0110 | IF $ACC \neq 0$ $PC = S$ |
| STOP | 0111 | stop |
| CALL S | 1000 | $SP = SP-1, [SP] = PC, PC = S$ |
| RETURN | 1001 | $PC = [SP], SP = SP + 1$ |
| PUSH | 1010 | $SP = SP-1, [SP] = ACC$ |
| POP | 1011 | $ACC = [SP], SP = SP + 1$ |
| LDR S | 1100 | $ACC = [[S]]$ |
| STR S | 1101 | $[[S]] = ACC$ |
| MOV PC | 1110 | $PC = ACC$ |
| MOV SP | 1111 | $SP = ACC$ |

Der Befehl Push



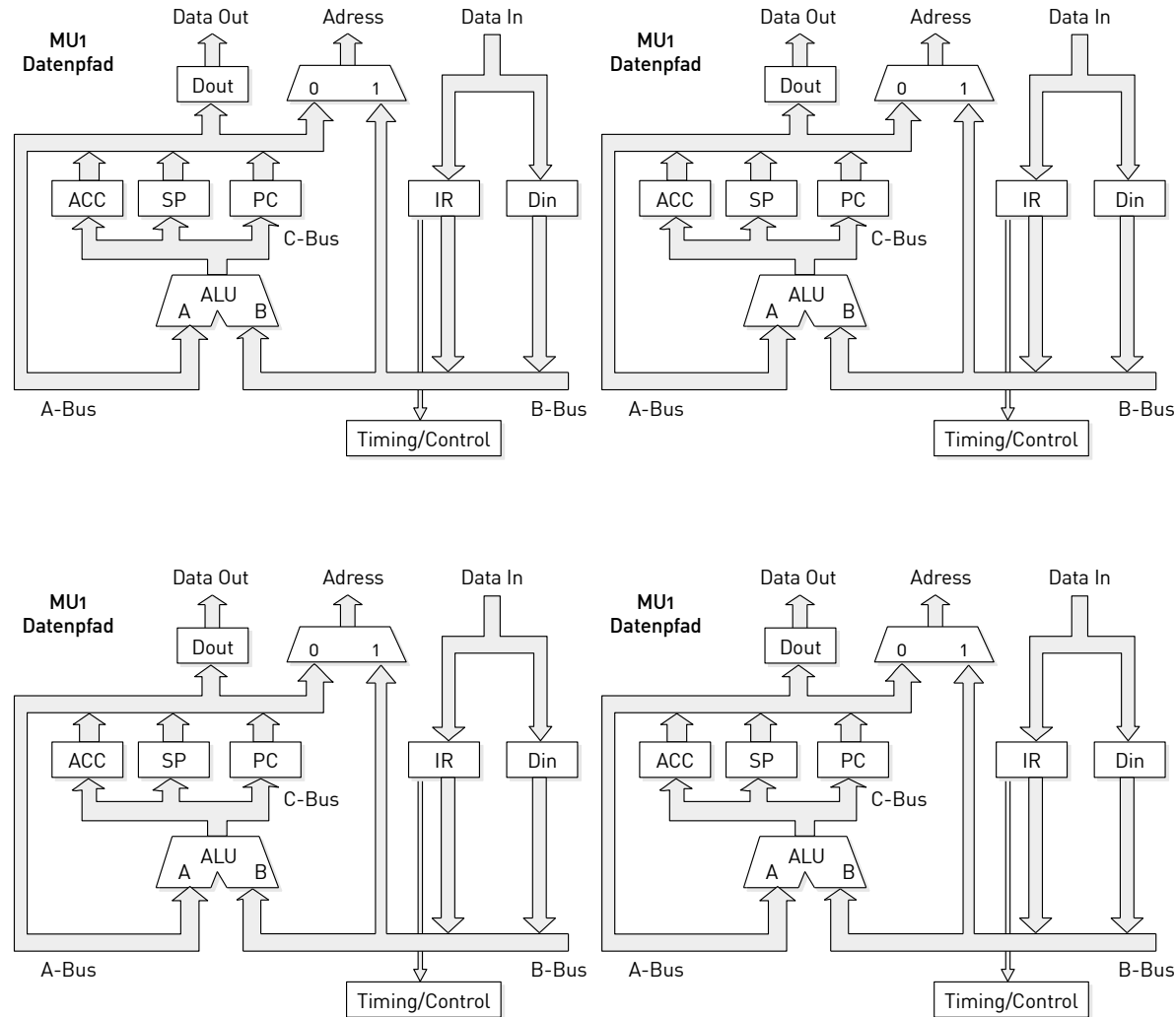
| Inputs | | | | | | Outputs | | | | | | | | | | | Description | | | | | | | |
|-------------|--------|--------|------|------------------------|----------------------------|---------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|-------------------|-----|--|--|
| Instruction | Opcode | /Reset | Step | ACC _Z /Zero | ACC ₁₅ /Negativ | Step | Address | ACC _{0E} | ACC _{1E} | SP _{oe} | SP _{ie} | PC _{oe} | PC _{ie} | IR _{oe} | IR _{ie} | DIN _{oe} | DIN _{ie} | DOUT _{oe} | DOUT _{ie} | ALU Function | MEM _{rq} | RnW | | |
| PUSH | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |

Der Befehl Pop



| Inputs | | | | | | Outputs | | | | | | | | | | | | Description | | | | | | | |
|--------|-------------|--------|--------|------|-----------|-----------------------------|------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|--------------------|-----|--|--|
| POP | Instruction | Opcode | /Reset | Step | ACCz/Zero | ACC ₁₅ /Negative | Step | Address | ACC _{0E} | ACC _{1E} | SP _{oe} | SP _{1e} | PC _{oe} | PC _{1e} | IR _{oe} | IR _{1e} | DIN _{oe} | DIN _{1e} | DOUT _{oe} | DOUT _{1e} | ALU Function | MEM _{r,q} | RnW | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | |

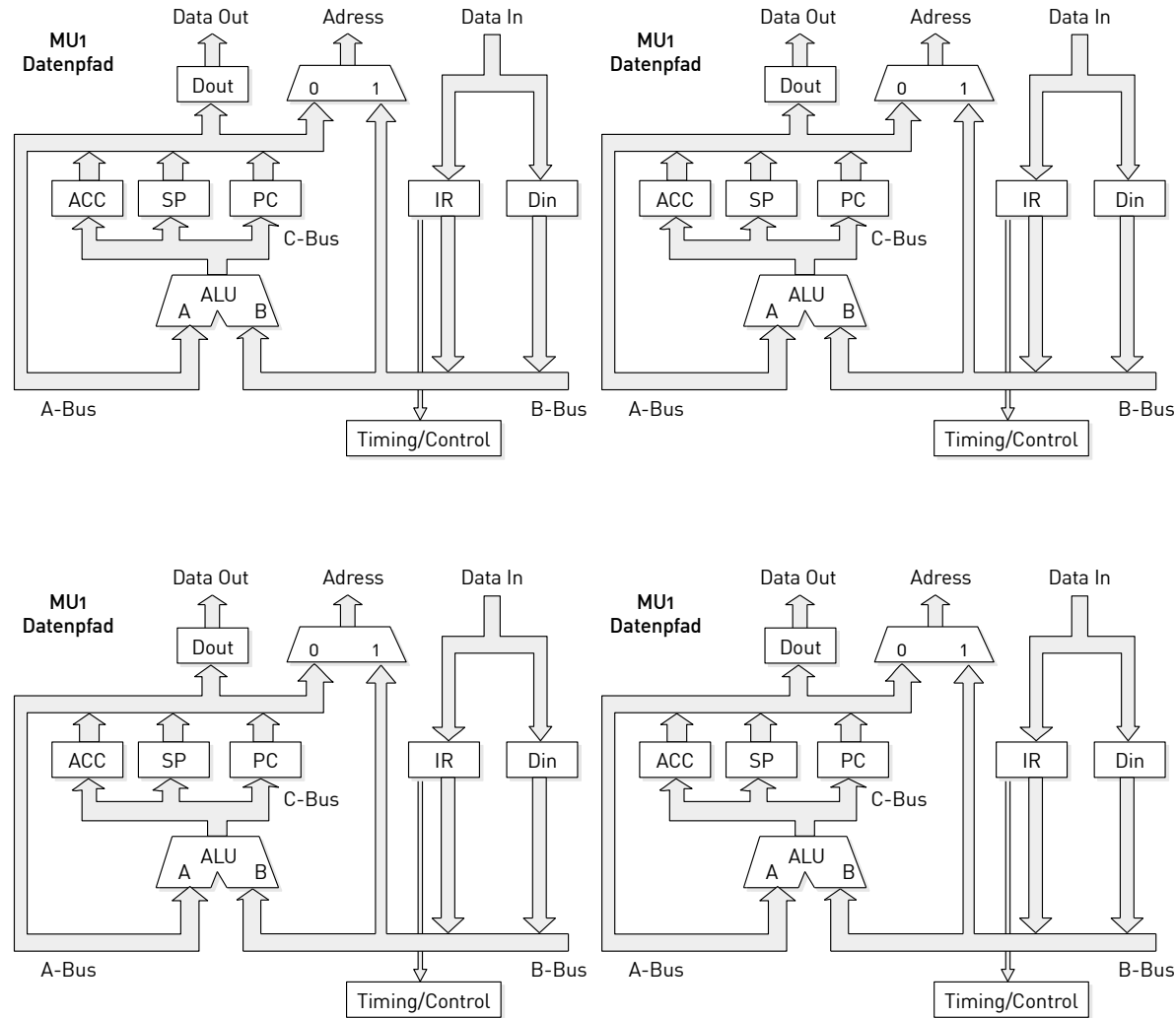
Der LDR S Befehl



Der LDR S Befehl

| Inputs | | Outputs | | | | | | | | | | | | | | | | Description | | | | | |
|-------------|--------|---------|------|-----------|-----------------------------|------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|-------------------|------|--|
| Instruction | Opcode | /Reset | Step | ACCz/Zero | ACC ₁₅ /Negative | Step | Address | ACC _{OE} | ACC _{IE} | SP _{oe} | SP _{ie} | PC _{oe} | PC _{ie} | IR _{oe} | IR _{ie} | DIN _{oe} | DIN _{ie} | DOUT _{oe} | DOUT _{ie} | ALU Function | MEM _{rq} | IRnW | |
| LDR S | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |

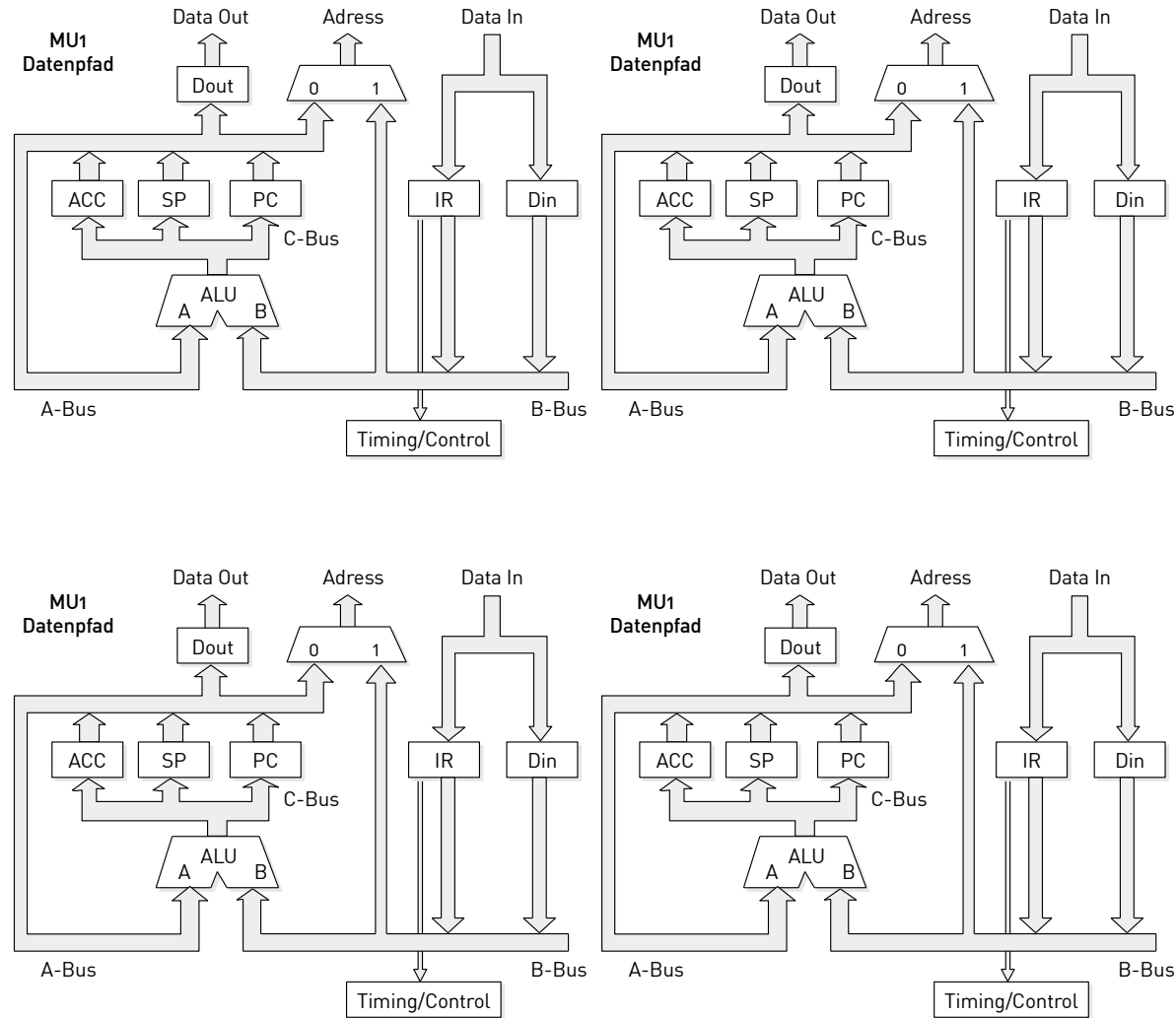
Der STR S Befehl



Der STR S Befehl

| Inputs | | Outputs | | | | | | | | | | | | | | Description | | | | | | | |
|-------------|--------|---------|------|-----------|-----------------------------|------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|--------------------|------|--|
| Instruction | Opcode | /Reset | Step | ACCz/Zero | ACC ₁₅ /Negative | Step | Address | ACC _{0E} | ACC _{1E} | SP _{0E} | SP _{1E} | PC _{0E} | PC _{1E} | IR _{0E} | IR _{1E} | DIN _{0E} | DIN _{1E} | DOUT _{0E} | DOUT _{1E} | ALU Function | MEM _{r,q} | IRnW | |
| STR S | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |

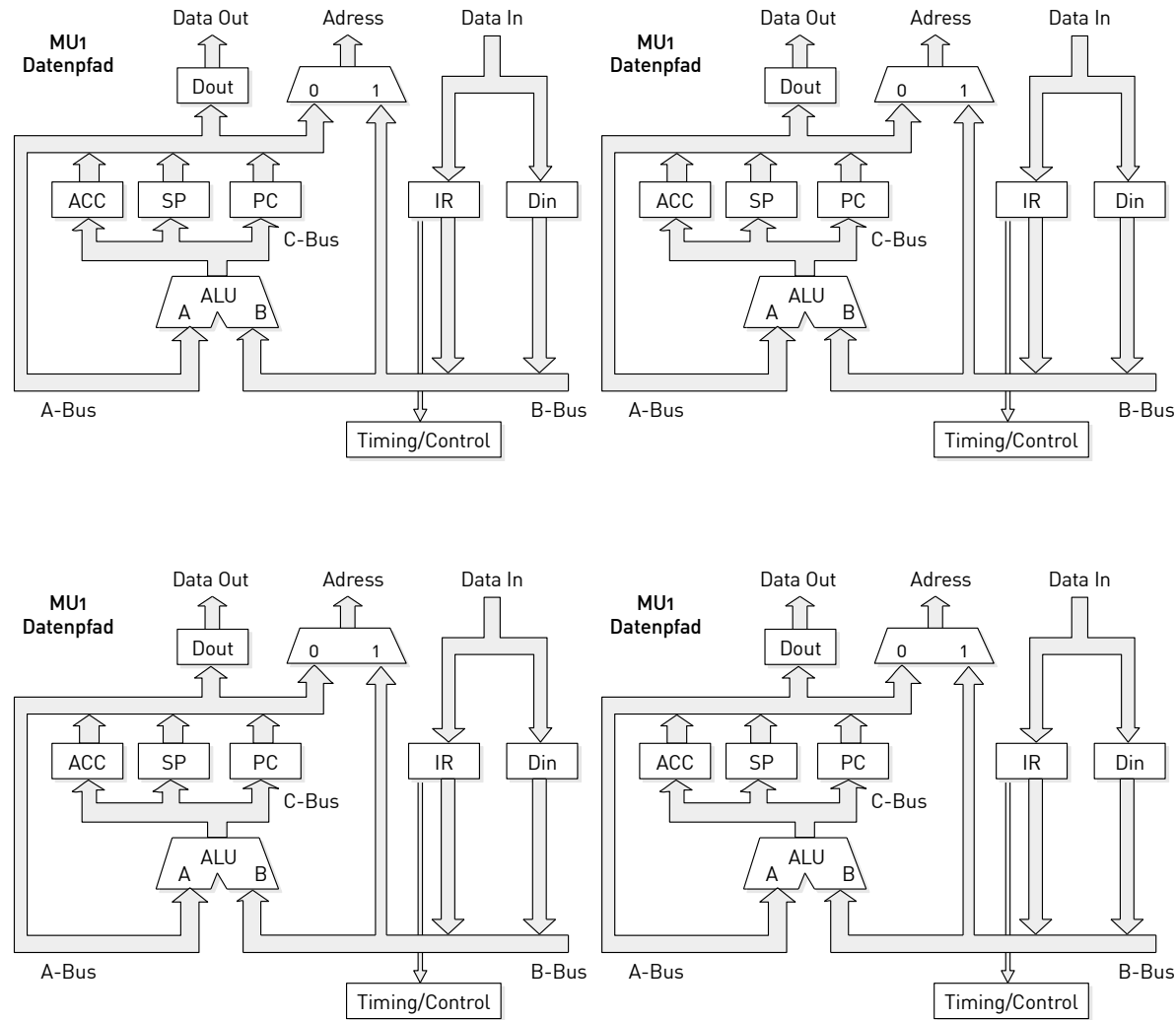
Der MOV PC Befehl



Der MOV PC Befehl

| Inputs | | | | | | Outputs | | | | | | | | | | | | Description | | | | | | |
|-------------|--------|--------|------|-----------|-----------------------------|---------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|--------------------|------|--|--|
| Instruction | Opcode | /Reset | Step | ACCz/Zero | ACC ₁₅ /Negative | Step | Address | ACC _{0E} | ACC _{1E} | SP _{0E} | SP _{1E} | PC _{0E} | PC _{1E} | IR _{0E} | IR _{1E} | DIN _{0E} | DIN _{1E} | DOUT _{0E} | DOUT _{1E} | ALU Function | MEM _{r,q} | IRnW | | |
| MOV PC | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |

Der MOV SP Befehl



Der MOV SP Befehl

| Inputs | | | | | | Outputs | | | | | | | | | | | | Description | | | | | | |
|-------------|--------|--------|------|------------------------|-----------------------------|---------|---------|-------------------|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|-------------------|--------------------|--------------------|--------------|-------------------|-----|--|--|
| Instruction | Opcode | /Reset | Step | ACC _Z /Zero | ACC ₁₅ /Negative | Step | Address | ACC _{0E} | ACC _{1E} | SP _{oe} | SP _{ie} | PC _{oe} | PC _{ie} | IR _{oe} | IR _{ie} | DIN _{oe} | DIN _{ie} | DOUT _{oe} | DOUT _{ie} | ALU Function | MEM _{rq} | RnW | | |
| MOV SP | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | |

Zusatzaufgabe (wenn in der Vorlesung behandelt):

Versuchen sie das Beispielprogramm aus der Vorlesung mit dem neuen Befehl LDR S so umzuschreiben, dass sie keinen selbst modifizierenden Code mehr benötigen.

```
Loop:      LDA  Total      ; Accumulate total
Add_instr: ADD  Table      ; Begin at head of table
           STO  Total      ;
           LDA  Add_instr  ; Change address ...
           ADD  One        ; by modifying instruction!
           STO  Add_instr  ;
           LDA  Count      ; Count iterations
           SUB  One        ; Count down to zero
           STO  Count      ;
           JGE  Loop       ; If >= 0 repeat
           STP                ; Halt execution
```

; Data definitions

```
Total      DEFW      0      ; Total - initially zero
One         DEFW      1      ; The number one
Count DEFW      4          ; Loop counter (loop 5x)
Table      DEFW      39     ; The numbers to total ...
           DEFW      25     ;
           DEFW      4      ;
           DEFW      98     ;
           DEFW      17     ;
```

Das obige Programm können Sie auch mit dem dem angebotenen Simulator (Ordner Simulator-MU1) testen. Wechseln hierzu in den Ordner und geben Sie in eine Konsole (Terminal)

```
java -jar ArchitectureSimulator_v8.jar
```

ein.

Infos hierzu finden sich im TechnoWiki vom Fachbereich Informatik unter:

https://wiki.h-da.de/fbi/technische-systeme/index.php/Projekt_mu0-Simulator