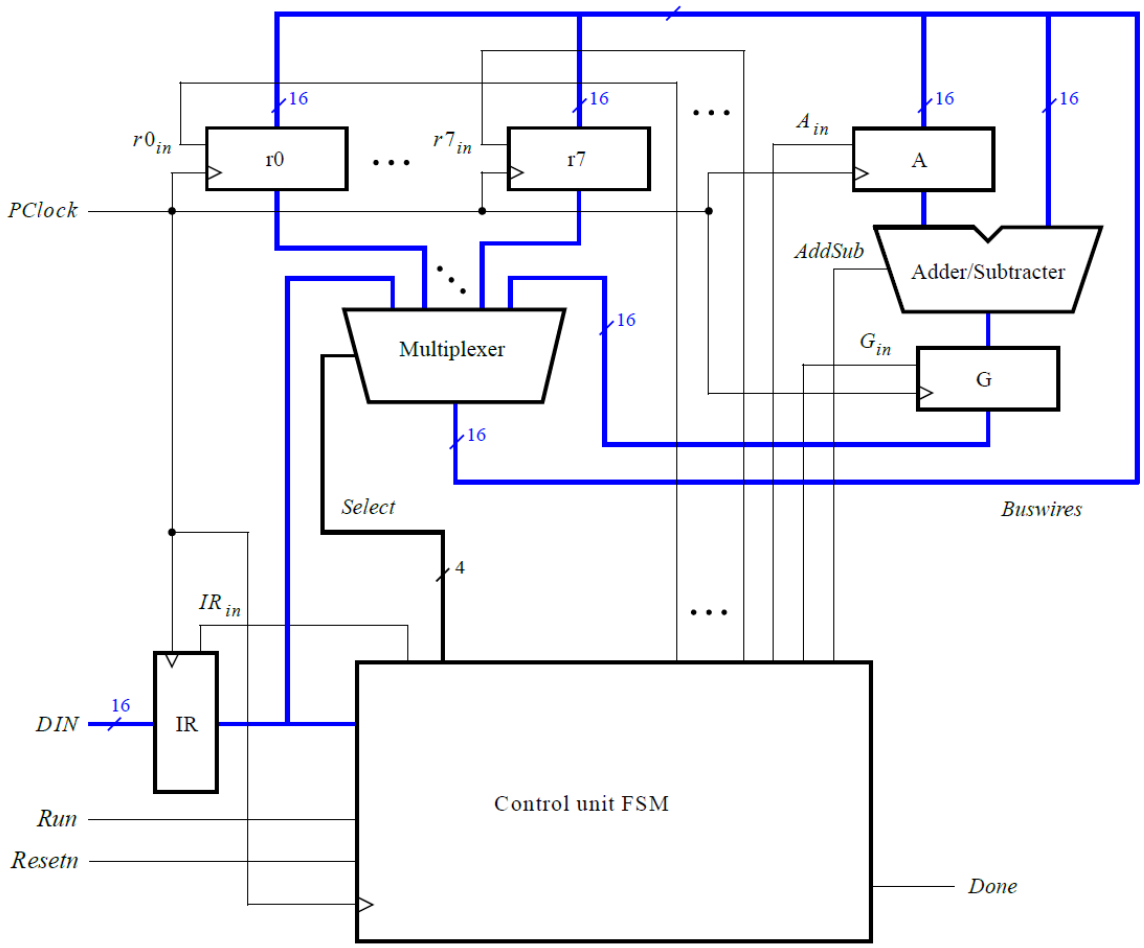
DE0_CV_simProc

DE0_CV_simProc ist eine einfache Implementation eines digitalen Systems mit einigen 16bit-Registern, einem Multiplexer, einem Addier-Subtrahierwerk und einer FiniteStateMachine als Ablaufsteuerung und stellt somit einen einfachen Prozessor dar.



Die Eingabe der Daten erfolgt über den Eingang DIN mit 16bit Breite. Diese Daten werden im Register IR zwischengespeichert. Mit Hilfe des Multiplexers können Daten von beliebigen Registern in jedes andere beliebige Register transferiert werden, z.B. vom IR in die *general purpose register* r0 – r7. Der Ausgang des Multiplexers steht an den Leitungen *BusWires* zur Verfügung. Die FSM steuert dabei die *Select* Lines, die die angesprochenen Register freischalten.

Das System kann mit jedem *Clock* cycle einfache Operationen durchführen:

Instruction		Function performed
mv	$rX, Op2$	$rX \leftarrow Op2$
mvt	$rX, \#D$	$rX_{15-8} \leftarrow D_{15-8}$
add	$rX, Op2$	$rX \leftarrow rX + Op2$
sub	$rX, Op2$	$rX \leftarrow rX - Op2$

Die Instruktionen werden über DIN in das IR geladen. Die Instruktionen sind 16bit lang und folgendermaßen codiert:

OPCODE format: III M XXX DDDDDDDDD

III = instruction, M = Immediate, XXX = rX.

wenn M = 0, DDDDDDDDD = 000000YYY = rY

wenn M = 1, DDDDDDDDD = #D ist der Operand

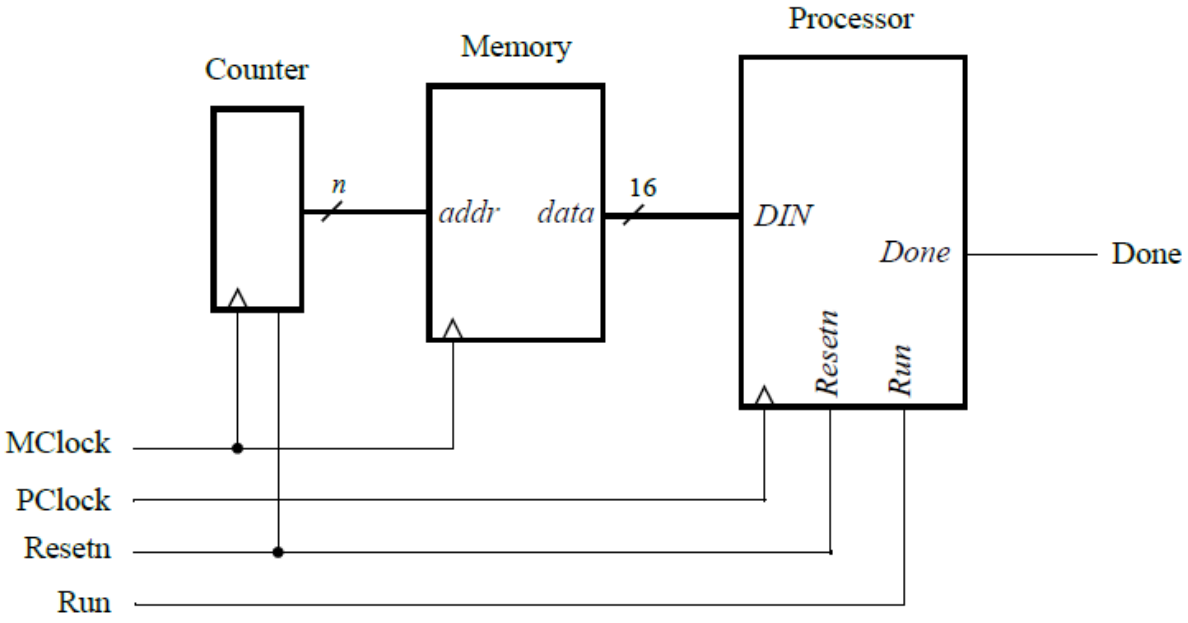
III	M	Instruction		Description
000	0	mv	rX,rY	rX <= rY
000	1	mv	rX,#D	rX <= D (0 extended)
001	1	mvt	rX,#D	rX <= D << 8
010	0	add	rX,rY	rX <= rX + rY
010	1	add	rX,#D	rX <= rX + D
011	0	sub	rX,rY	rX <= rX – rY
011	1	sub	rX,#D	rX <= rX - D

Die Anzahl der dafür notwendigen Clock cycles ist unterschiedlich:

Die Tabelle zeigt die Anzahl der Befehlsschritte und die darin durchgeführten Aktionen.

	T_0	T_1	T_2	T_3
mv	IR_{in}	*Select* = *rY* or *IR*, rX_{in}, *Done*		
mvt	IR_{in}	*Select* = *IR*, rX_{in}, *Done*		
add	IR_{in}	*Select* = *rX*, A_{in}	*Select* = *rY* or *IR*, G_{in}	*Select* = *G*, rX_{in}, *Done*
sub	IR_{in}	*Select* = *rX*, A_{in}	*Select* = *rY* or *IR*, *AddSub*, G_{in}	*Select* = *G*, rX_{in}, *Done*

Mit dem MClock (KEY0) wird der nächste Befehl aus dem Speicher geholt.



Der PClock dient zum Abarbeiten der Prozessorzyklen.

Berechnen des Zweierkomplements einer Zahl

Zum Austesten unseres Systems werden wir das Einer- und Zweierkomplement einer vorgegebenen Zahl berechnen.

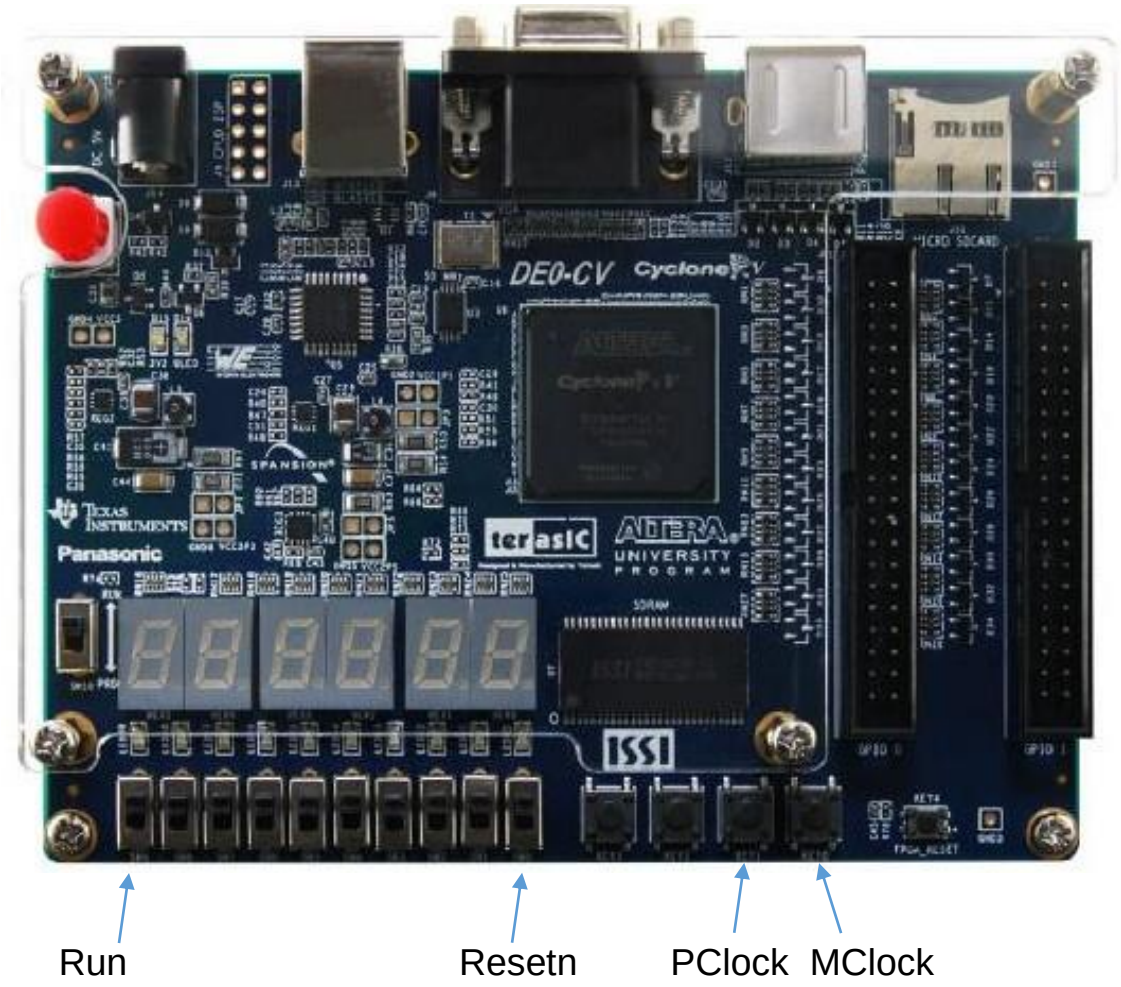
Wir verwenden dazu folgenden Code:

```
mv    r0, #28          //    Zahl = 28d = 1Ch
mvt   r1, #0xFF00      //
add   r1, #0x00FF      //    r1 = 0xFFFF
sub   r1, r0           //    r1 => Einerkomplement von r0
add   r1, #1           //    r1 => Zweierkomplement von r0
```

Im Speicher ist dieses Programm folgendermaßen hinterlegt:

Zeile	**III**	**M**	**XXX**	**D DDDD DDDD**
0	000	1	000	0 0001 1100
1	001	1	001	0 1111 1111
2	010	1	001	0 1111 1111
3	011	0	001	0 0000 0000
4	010	1	001	0 0000 0001

Downloaden des VHDL-Projekts



MClock	Memory Clock
PClock	Prozessor Clock

An den vier rechten Sieben-Segment-Anzeigen wird der Inhalt von *BusWires* angezeigt (HEX3 – HEX0).

In der ganz linken Anzeige sieht man den Inhalt des Programm-Counters (HEX5).

Ablauf zum Testen:

- sof-File downloaden
- alle Switches ausschalten
- 1x Betätigen von MClock (KEY0)
- Run einschalten (SW9), LEDR9 leuchtet
- Resetn aufheben (SW0)
- 1x MClock (Befehl holen)
- PClock (KEY1) solange betätigen bis die LEDR0 leuchtet (Done)
- 1x MClock
- PClock solange betätigen bis die LEDR0 leuchtet (Done)
- 1xMClock
- …

Durch Drücken und Halten von KEY3 kann der Inhalt von Register A betrachtet werden.