



h_da

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbi
FACHBEREICH INFORMATIK

RECHNERARCHITEKTUR

SS17

Termin2

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

Bereiten Sie die Lösungen daheim so vor, dass Sie die Ergebnisse in der Übung präsentieren können.

Aufgabe:

Erweitern sie den Befehlssatz des MU1 Prozessors um die Befehle LDR16 , STR, CALL16 und IADD. Malen Sie in die Diagramme den jeweiligen Datenfluss. Überlegen sie sich den Datentransport mit Hilfe der Datenflußbilder und versuchen sie die Befehle mit Hilfe der Microcode Befehle zu beschreiben. Wenn sie den Befehl verstanden haben, füllen sie die Steuerungstabelle aus. Führen sie dies in Ihrer Vorbereitung auf das Praktikum aus, da wir im Praktikum über Ihre Ergebnisse diskutieren möchten

Der Befehl LDR16 lädt den Wert einer 16 Bit Adresse, die im Programm direkt hinter dem Opcode steht, über das Din Register in den Akkumulator. Sie können das Din Register zuerst als Hilfsregister benutzen um die Adresse aus dem Programmcode zu laden. Vergessen sie nach dem Programmzugriff nicht, dass der Programcounter danach wieder auf den nächsten Befehl zeigen muss. (Lösung in 2 Takten)

Der Befehl STR S speichert den Inhalt des Akkumulator auf die Adresse, auf den der Inhalt der Variablen S (steht im IR) zeigt. Sie brauchen also zwei Speicherzugriffe. Die Variable S ist in diesem Fall Bestandteil des Opcodes und befindet sich in den unteren 12 Bit des Befehlswords. (Lösung in 2 Takten)

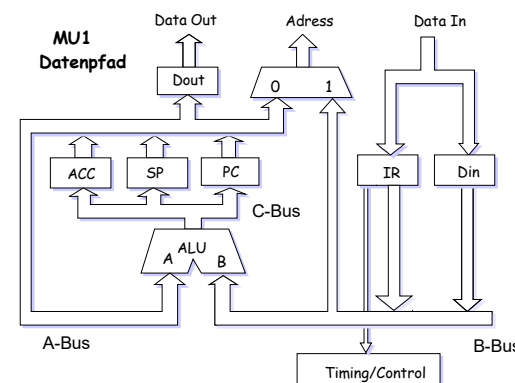
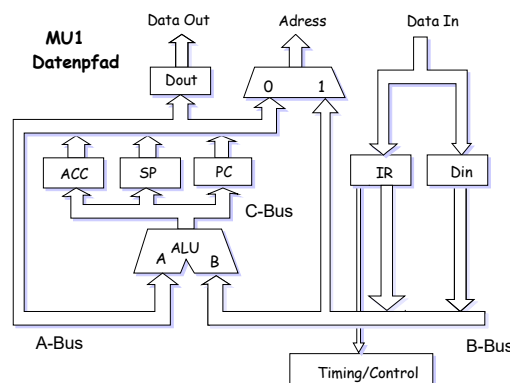
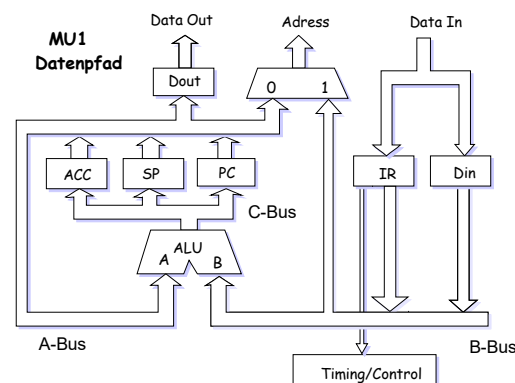
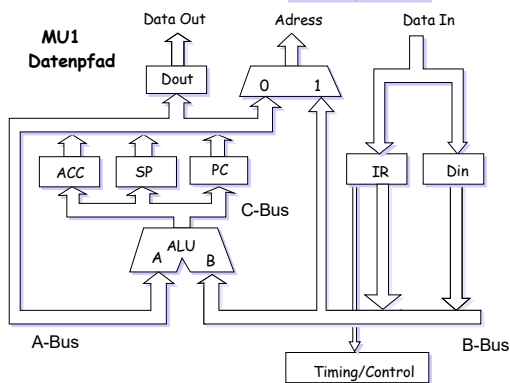
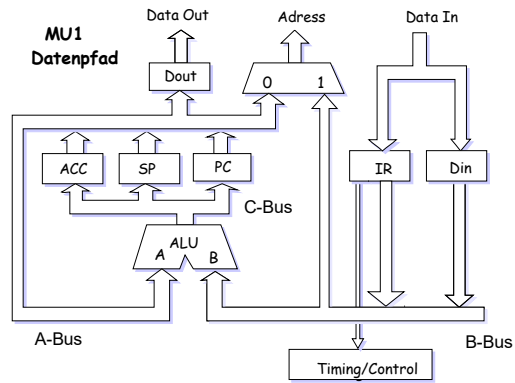
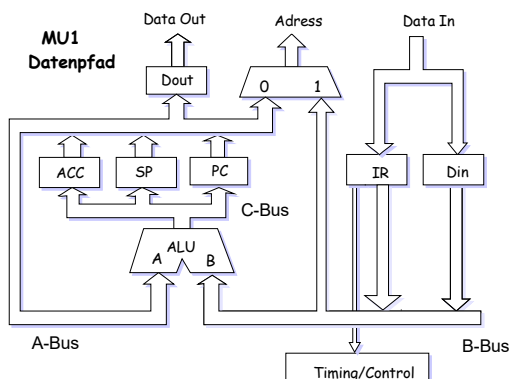
Der Befehl CALL16 S führt einen Unterprogramm Aufruf durch. Der aktuelle Programcounter wird auf dem Stack gespeichert und der Wert S, der direkt hinter dem Opcode im Programm steht, wird als neuer Programcounter benutzt. (Lösung in 4 Takten)

Der Befehl IADD addiert die beiden obersten (physikalisch die untersten) Stackwerte und schreibt das Ergebnis auf den Stack zurück. Der Stackpointer ist nach dieser Operation um eins erhöht. (Lösung in 5 Takten)

Befehlstabelle für Mu1

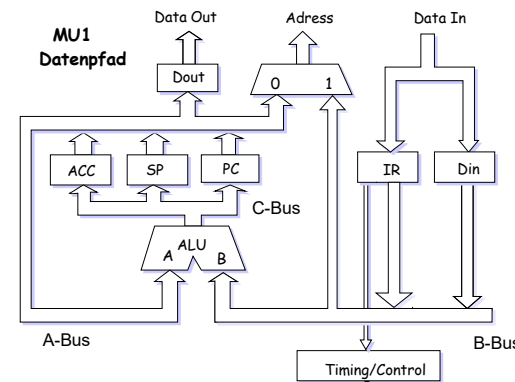
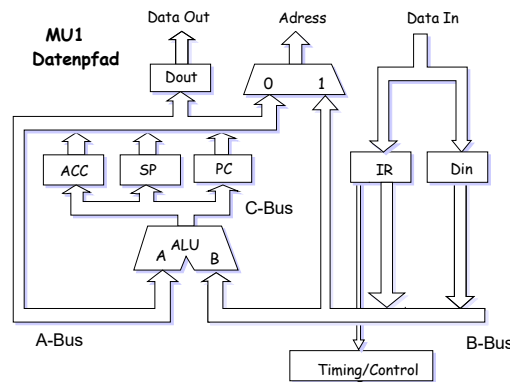
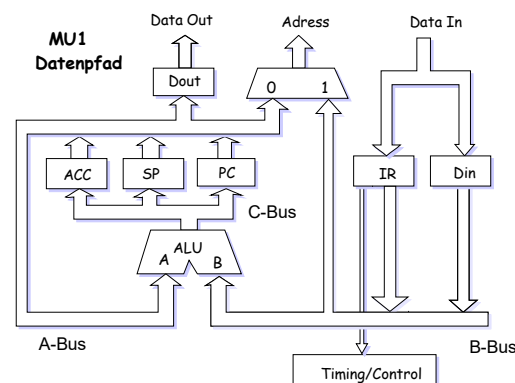
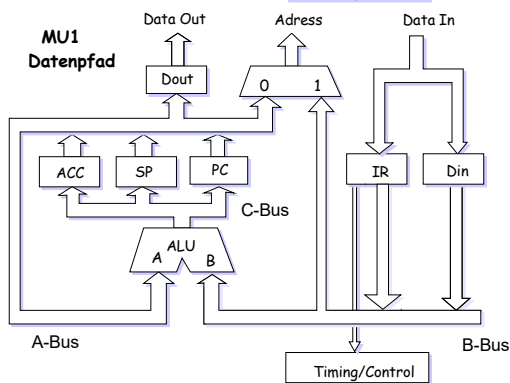
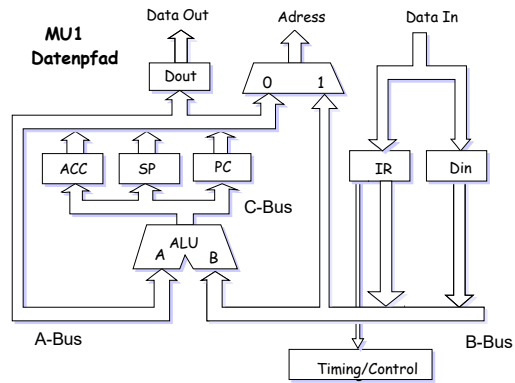
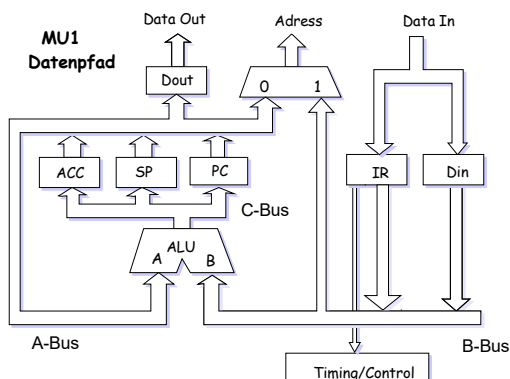
<i>Instruktion</i>	<i>Effekt</i>	<i>Instruktion</i>	<i>Effekt</i>
Reset	PC = 0	LDR16 S	
LDA S	Din = [IR]; A = Din	CALL16 S	
STO S	Dout = A; [IR] = Dout	RETURN	Din = [SP], SP = SP + 1; PC = Din
ADD S	Din = [IR], A = A + Din	PUSH	SP = SP - 1; Dout = A; [SP] = Dout
SUB S	Din = [IR]; A = A - Din	POP	Din = SP, SP = SP + 1; A = [SP]
JMP S	Din = [IR]; PC = Din	LDR S	Din = [IR]; Din=[Din], Dout = A; [Din] = Dout
JGE S	IF !N => PC = IR	STR S	
JNE S	IF !Z => PC = IR	IADD	
STOP	stop		

Der Befehl LDR16 S



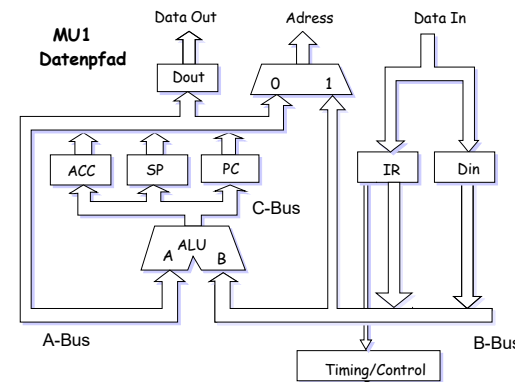
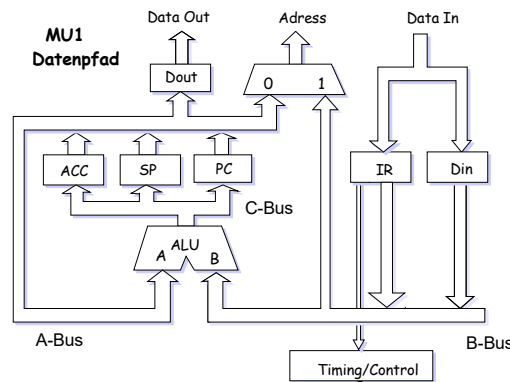
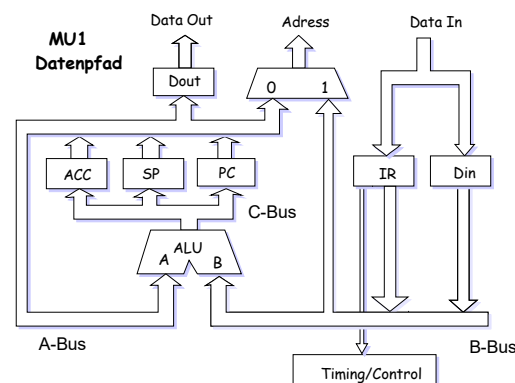
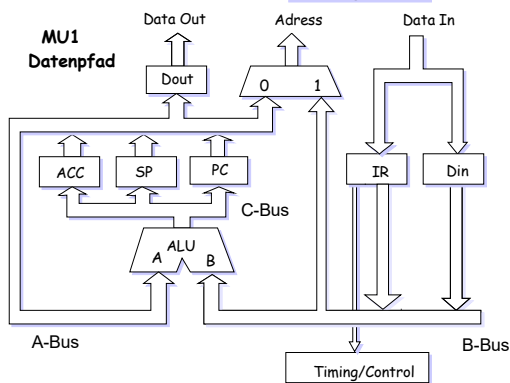
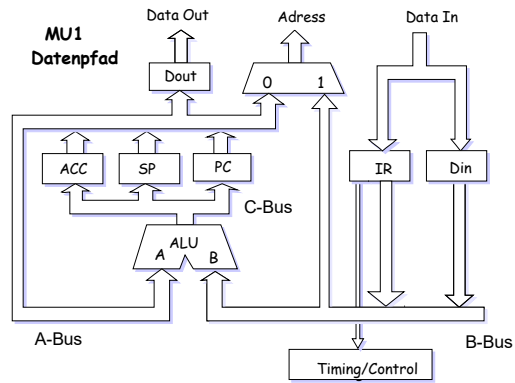
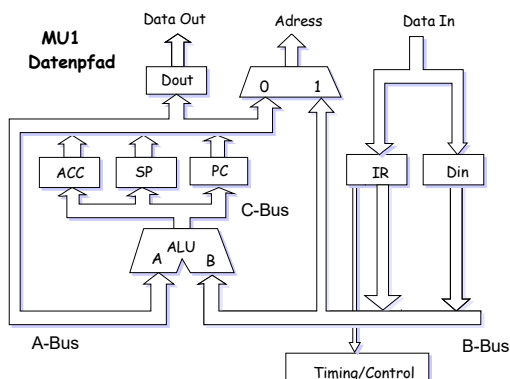
Inputs						Outputs																
Instruction	Opcode	/Reset	step	Z	N	step	Address	Aoe	Aie	Pcoe	Pcie	Iroe	Irie	Spoe	Spie	DINoe	DINie	DOUToe	DOUTie	ALU Function	MEMrq	RnW
LDR16 S																						

Der Befehl STR S



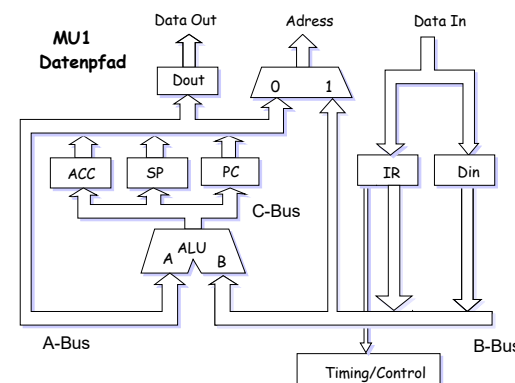
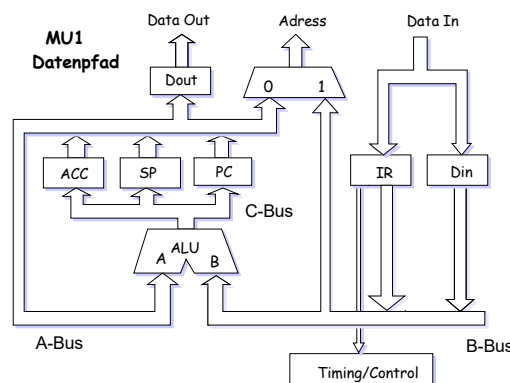
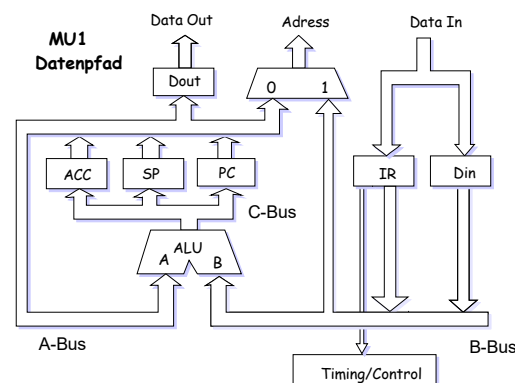
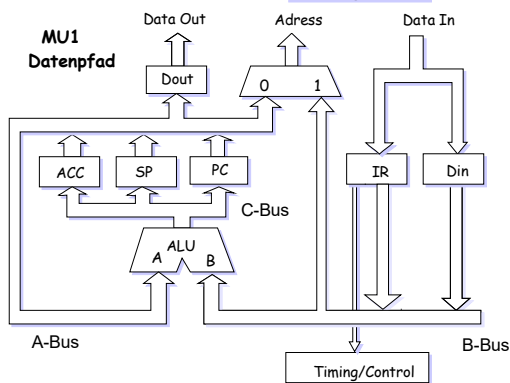
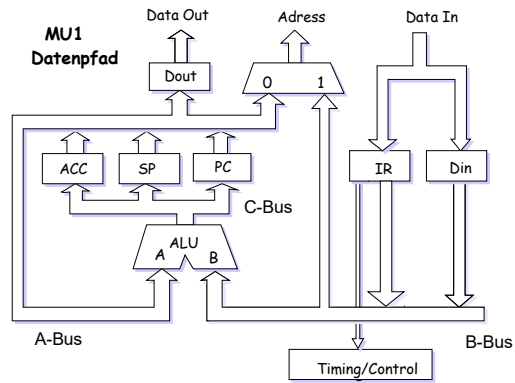
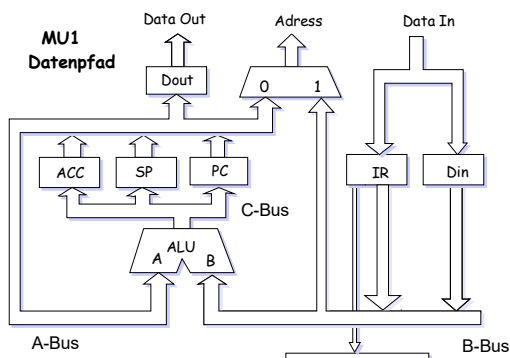
Inputs					Outputs																	
Instruction	Opcode	/Reset	step	Z	N	step	Address	Aoe	Aie	Pcoe	Pcie	Iroe	Irie	Spoe	Spie	DINoe	DINie	DOUToe	DOUTie	ALU Function	MEMIrq	RnW
STR S																						

Der CALL16 S Befehl



Inputs					Outputs																	
Instruction	Opcode	/Reset	step	Z	N	step	Address	Aoe	Aie	Pcoe	Pcie	Iroe	Irie	Spoe	Spie	DINoe	DINie	DOUToe	DOUTie	ALU Function	MEMrq	Rnw
CALL16 S																						

Der IADD Befehl



Inputs		Outputs																				
Instruction	Opcode	/Reset	step	Z	N	step	Address	Aoe	Aie	Pcoe	Pcie	Iroe	Irie	Spoie	Spie	DInoe	DINie	DOUToe	DOUtie	ALU Function	MEMrq	RnW
IADD																						