Kapitel 7

Prozessor-Grundlagen

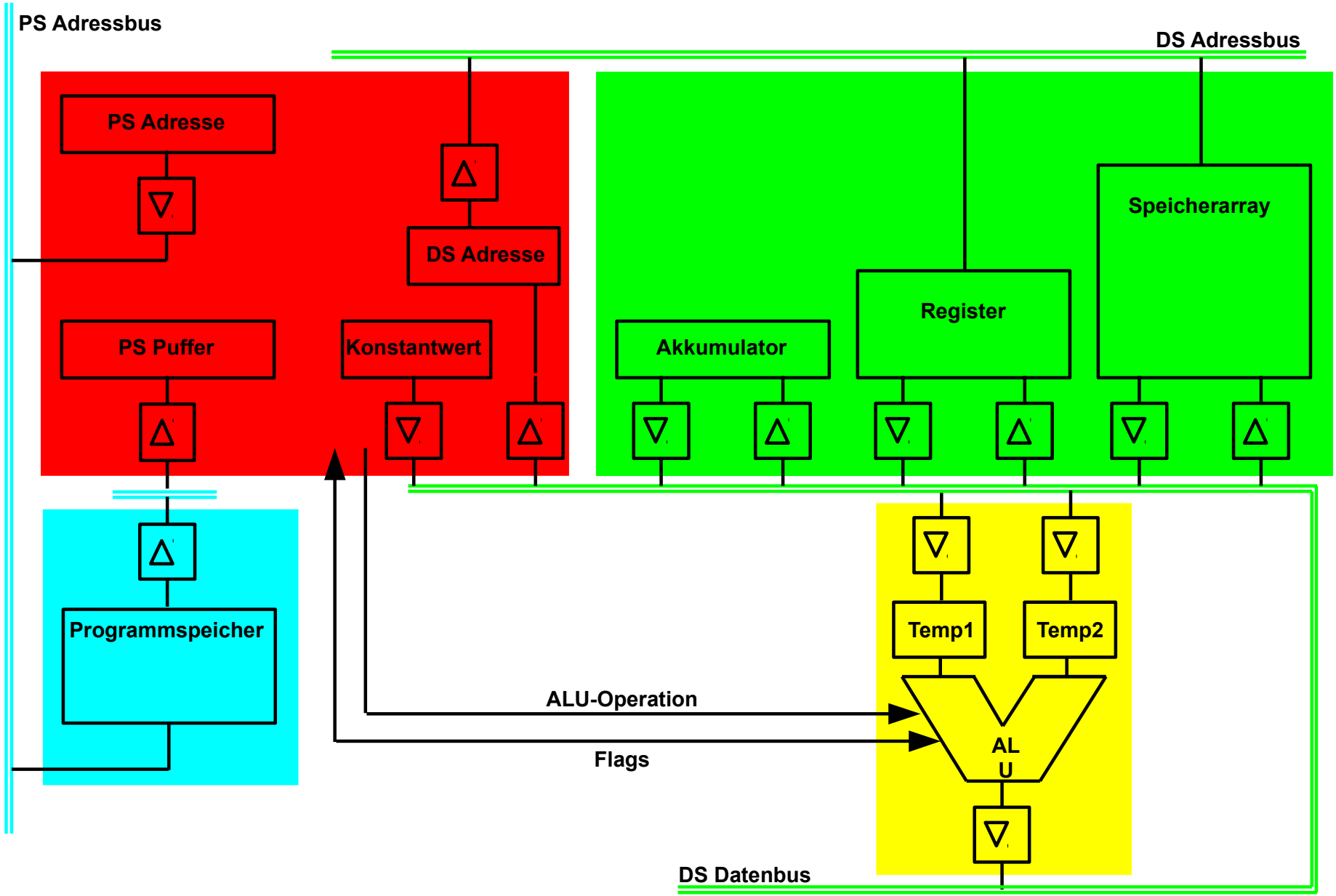
Prozessor Grundlagen

Funktionsweise des Steuerwerks

Harvard Architektur

Mehr Details zeigt die folgende Folie :

8051 Prozessor Grundstruktur



Programmspeicher

- Bei schwächeren Prozessoren meist ein ROM
  - z.B. 8051

Datenspeicher

- Der in der Architektur aufgeführte **Datenspeicher** beinhaltet im Prinzip
  - Speicherarray
    - DRAM/SRAM oder im Prozessorchip integriert
    - große Kapazität
    - langsam
  - Register
    - in Prozessorschaltkreisen integriert
    - kleine Anzahl (8, 16 oder seltener 32)
    - schnell
  - Akkumulator (nicht immer vorhanden)
    - in Prozessorschaltkreisen integriert
    - ein Datenwort groß
    - vereinfacht den Design

Rechenwerk : Arithmetic and Logic Unit (ALU)

- ALU Operation : z.B. ADD (x + y → z)
- Die beiden Operanden und das Ergebnis residieren im Speicherarray, in Registern oder im Akkumulator
- Unter Kontrolle des Steuerwerks folgende Operationen durchführen
  1. einen Operanden (x) über den Datenbus ins Register Temp1 übernehmen,
  2. anderen Operanden (y) über den Datenbus ins Register Temp2 übernehmen,
  3. die Spezifikation der durchzuführenden Operation vom Steuerwerk empfangen,
  4. ALU-Operation (z.B. ADD) in der ALU durchführen,
  5. Ergebnis (z) am Datenbus zur Verfügung stellen,
  6. Flag Werte (z.B. Carry, ALU = 0, Overflow) ans Steuerwerk melden.

Steuerwerk

- Das Steuerwerk steuert die Ausführung der Befehle
  - Das Steuerwerk holt einen OpCode aus dem Programmspeicher (siehe unten)
  - Das Steuerwerk dekodiert den Befehl
    - Das Steuerwerk identifiziert die durchzuführende Operation
    - Das Steuerwerk identifiziert die an der Operation beteiligten Operanden
    - Das Steuerwerk isoliert die Operandenadressen
  - Das Steuerwerk führt den Befehl aus
    - Das Steuerwerk steuert den Transfer der Operanden zum Rechenwerk (siehe unten).
    - Das Steuerwerk teilt dem Rechenwerk den Typ der durchzuführenden Operation mit.
    - Das Steuerwerk steuert den Transfer des Ergebnisses.
  - Das Steuerwerk modifiziert das Register PS Adresse
    - Das Steuerwerk inkrementiert das Register PS Adresse nach jedem Programmspeicherzugriff.
    - Das Steuerwerk lädt eine neue Adresse bei einem Sprungbefehl.

Steuerwerk (Forts.)

- Steuert die Datenübertragung über die Systembusse
  - Um einen Befehl aus dem Programmspeicher zu holen,
    - Lädt das Steuerwerk den Inhalt des Befehlszeigers in das Register **PS Adresse** und aktiviert das zugehörige OUT-Gatter.
    - Der Programmspeicher stellt den Befehl auf seinem Datenbus zur Verfügung.
    - Das Steuerwerk aktiviert das IN-Gatter zum Register PS Puffer.
    - Jetzt kann das Steuerwerk den Befehl dekodieren und ausführen.
    - Wenn der Befehl aus mehreren Bytes besteht, holt das Steuerwerk die weiteren Bytes in gleicher Weise über das Register PS Puffer.
  - Das Steuerwerk steuert die Datenübertragung über den Datenbus.
    - Befindet sich ein Operand oder das Ergebnis im Speicherarray oder einem Register, lädt das Steuerwerk die entsprechende Adresse in das Register **DS Adresse.**
    - Befindet sich ein Operand oder das Ergebnis im Akkumulator ist keine Ansteuerung von Adressleitungen notwendig.
    - Für die Übertragung zur ALU aktiviert das Steuerwerk das OUT-Gatter von Speicherarray, Register, oder Akkumulator **und** das IN-Gatter von Temp1 oder Temp2.
    - Für die Übertragung aus der ALU aktiviert das Steuerwerk das OUT-Gatter der ALU und das IN-Gatter von Speicherarray, Register, oder Akkumulator.
    - Ein Transportbefehl (z.B.: von einem Register in das Speicherarray) steuert das Steuerwerk durch entsprechende Aktivierung der OUT- und IN-Gatter von Akkumulator, Registern und Speicherarray.

Timing

- Ein Maschinenbefehl ist aus Teiloperationen zusammen gesetzt, die in mehreren aufeinander folgenden Schritten durchgeführt werden.
- Ein zentraler Takt steuert die zeitgerechte Ausführung dieser Schritte.
- Ein Oszillator erzeugt ein rechteckförmiges Signal, (z.B. 10 MHz beim 80c515c)
- Mehrere Perioden des Oszillatorsignals bilden einen **Maschinenzyklus,**
  (z.B. beim 80c515c die 6 Takte S1, S2, …, S6)
- Die Ausführung jedes Befehl benötigt eine bestimmte Anzahl von Maschinenzyklen,
  (z.B. 1, 2, 3, oder 4 Zyklen beim 80c515c)

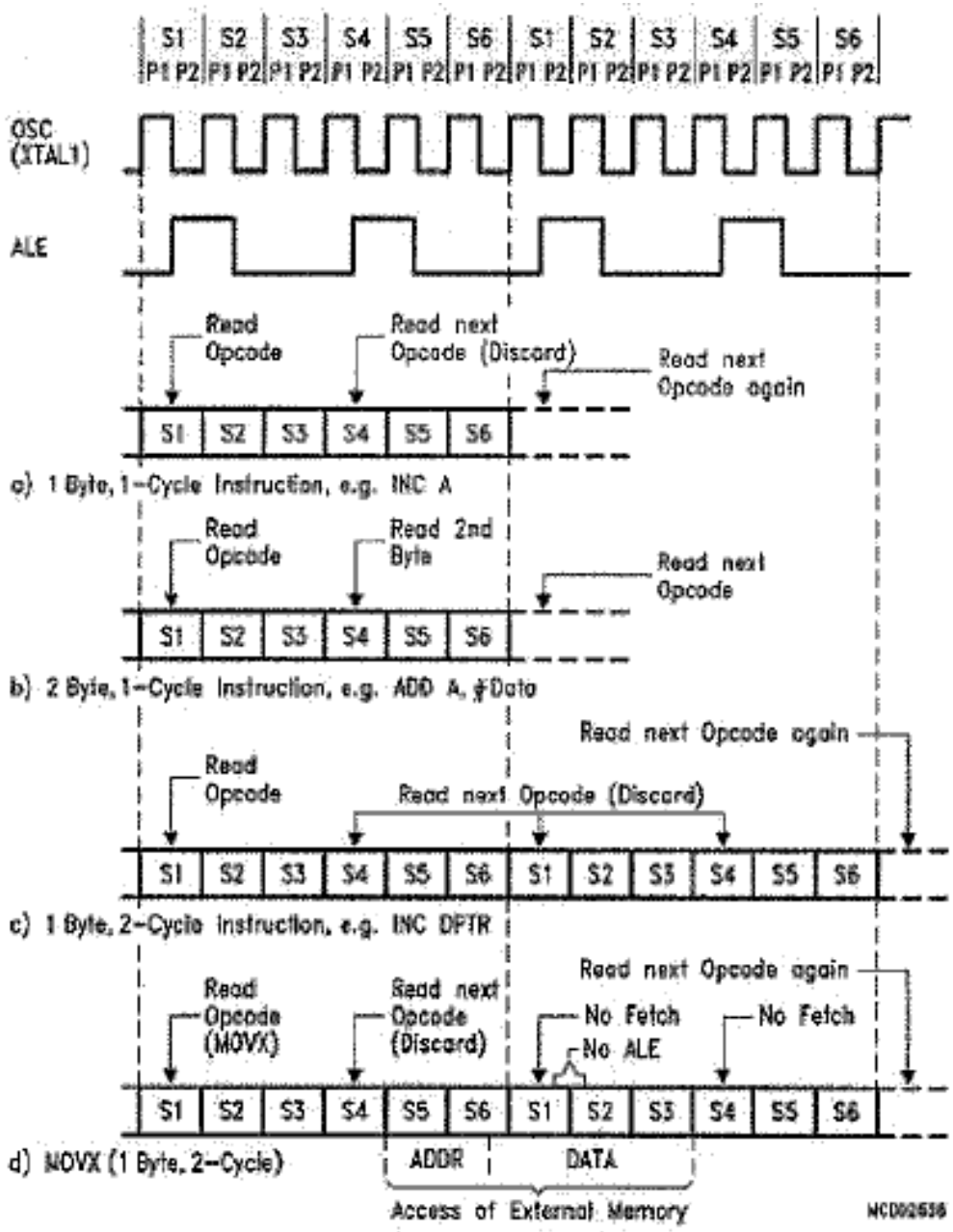


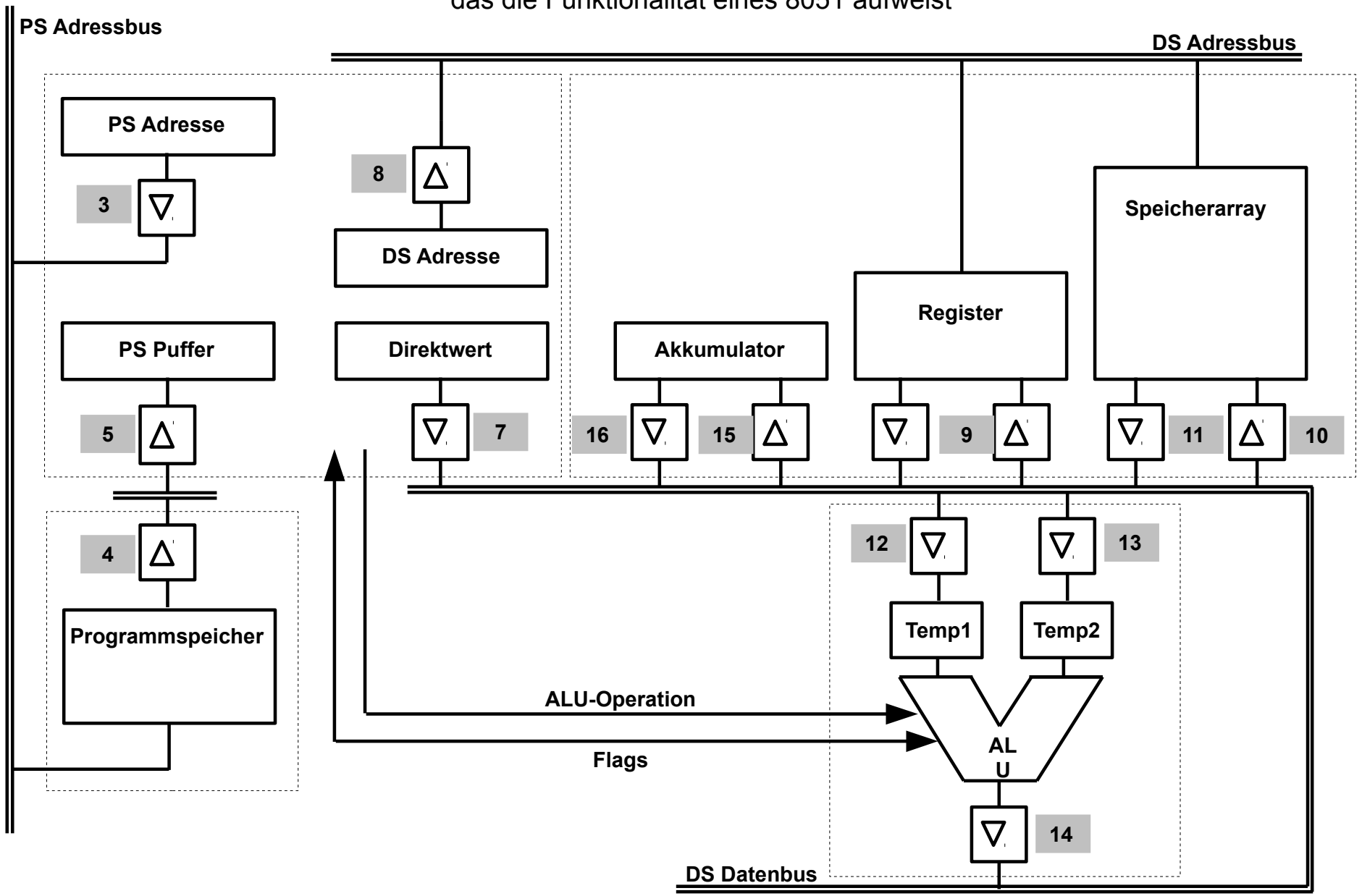
Bild 33: Befehlsausführung 80c515c

80c515c Timing

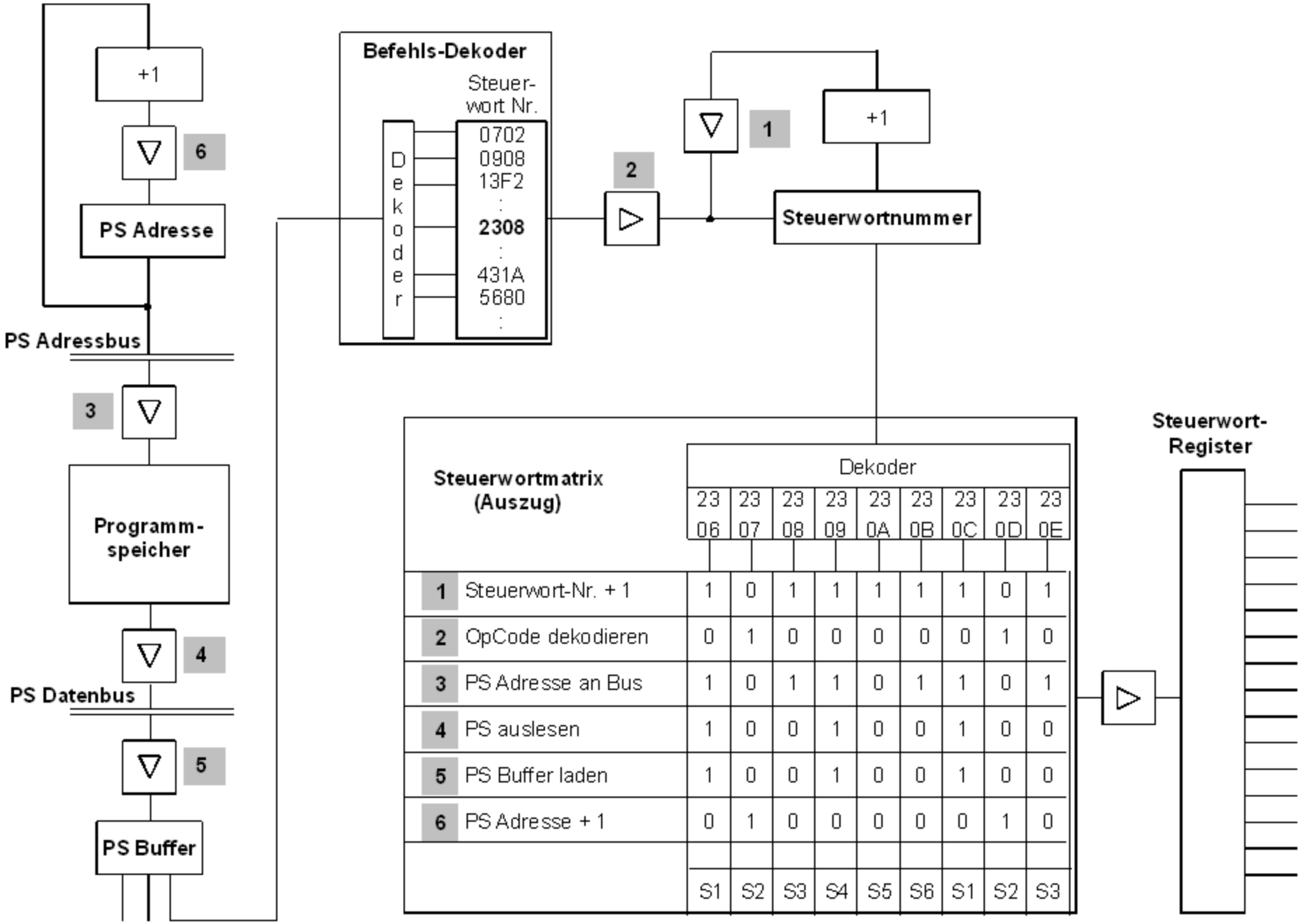
- Der 80c515c Prozessor liest in jedem Maschinenzyklus zwei Bytes aus dem Programmspeicher aus, je ein Byte im Step 1 und im Step 4.
- Wenn das ausgelesene Byte noch nicht benötigt wird, liest der Prozessor das selbe Byte drei Steps später erneut.
- Wenn ein ausgelesenes Byte tatsächlich verwendet wird, wird auch das Register PS Adresse inkrementiert.
- Ein **OpCode** wird immer im **Step 1** gelesen.
- Befehlstyp 1 Byte – 1 Zyklus (z.B. MOV A,R1)
  - Der ganze Befehl (1 Byte) wird im Step 1 ausgelesen und das Register PS Adresse inkrementiert
  - In Step 2 bis Step 6 wird der Befehl ausgeführt.
  - Das im Step 4 ausgelesene Byte (eigentlich der OpCode des nächsten Befehls) wird ignoriert.
- Befehlstyp 2 Byte – 1 Zyklus (z.B. ADD A,#const)
  - Der OpCode wird im Step 1 ausgelesen und das Register PS Adresse inkrementiert
  - Im Step 4 wird das zweite Byte des Befehls (im Beispiel der Immediate-Wert) ausgelesen und das Register PS Adresse erneut inkrementiert
  - In Step 2 bis Step 6 wird der Befehl ausgeführt.
- Bei Befehlstypen, die mehr als einen Zyklus benötigen, stehen für die Ausführung des Befehls Step 2 bis Step6 des ersten Zyklus und alle Steps der weiteren Zyklen zur Verfügung.

Ansatz für ein hypothetisches Steuerwerk

das die Funktionalität eines 8051 aufweist



Steuerung der Befehlsausführung



Steuerwortmatrix

- Ein ROM Array im Steuerwerk, das spezifiziert, welche Gatter und Funktionseinheiten zu welchem Zeitpunkt aktiviert werden.
  - 1 : Das Gatter oder die Funktionseinheit wird aktiviert.
  - 0 : Das Gatter oder die Funktionseinheit nicht wird aktiviert.
- Jede Zeile der Steuerwortmatrix spezifiziert für jeden Step, ob ein Gatter oder eine Funktionseinheit aktiviert wird oder nicht.
  - z.B. das Gatter **5**, das das Byte vom PS Datenbus in den PS Buffer übernimmt.
    - Zur Erinnerung : In jedem S1 und jedem S4 wird ein Byte aus dem Programmspeicher ausgelesen.
- Für jeden Step eines jeden Maschinenbefehls ist eine Spalte der Steuerwortmatrix, ein sog. **Steuerwort** vorgesehen.
- Eine Folge von 6, 12 oder 18 Steuerworten (Spalten) spezifiziert, welche Gatter für die Ausführung eines Maschinenbefehls aktiviert werden müssen.

Befehls-Dekoder

- Ein ROM Array im Steuerwerk mit 256 Einträgen (für die OpCodes 0x00 … 0xFF) zu je zwei Bytes
- Jeder Eintrag ist die Steuerwortnummer des ersten Steuerworts für den betreffenden Maschinenbefehl.

Betrieb des Prozessors

- In jedem Oszillatorzyklus
  1. lädt das Steuerwerk das durch das Register Steuerwortnummer adressierte Steuerwort in das Steuerwortregister,
  2. aktiviert die Gatter, die im Steuerwortregister eine 1 aufweisen.
- Mit jeder 1 im Gatter **1** (Steuerwort-Nr. + 1) wird zum nächsten Steuerwort weiter geschaltet → der Prozessor läuft.
- In jedem S1 und jedem S4 wird ein Byte aus dem Programmspeicher gelesen → siehe Gatter **3**, **4** und **5**
- Wenn in einem S1 ein OpCode aus dem Programmspeicher ausgelesen wurde
  - steht im Gatter **1** (Steuerwort-Nr. + 1) der nachfolgenden Spalte eine 0,
  - im Gatter **2** (OpCode dekodieren) steht dann aber eine 1
    - Der Wert im Register Steuerwortnummer wird ersetzt durch die Steuerwortnummer von S3 für den gerade gelesenen OpCode
- Eigentlich ist S3 der erste Step bei der Ausführung eines Befehls. Bei einem Befehl, der einen Maschinenzyklus dauert, wird im darauf folgenden S1 und S2 der nächste OpCode gelesen und dekodiert.
- S1 und S2, die den nächsten OpCode lesen und dekodieren, sind die letzten Steps einer Befehlsausführung.
- Das Laden des Registers Steuerwortnummer über das Gatter **2** ist sozusagen eine **n-way Verzweigung** zum Steuerwort von S3 des als nächstes auszuführenden Befehls.

Steuerwortmatrix (Auszug)

	Steuerwort-Nr. High	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23	23
	Steuerwort-Nr. Low	**08**	**09**	**0A**	**0B**	**0C**	**0D**	**0E**	**0F**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**	**1B**	**1C**	**1D**	**1E**	**1F**
		S3	S4	S5	S6	S1	S2	S3	S4	S5	S6	S1	S2	S3	S4	S5	S6	S1	S2	S3	S4	S5	S6	S1	S2
1	Steuerwort-Nr.+1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0
2	Befehl dekodieren	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
3	PS Adresse an Bus	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
4	PS auslesen	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0
5	PS Buffer laden	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0
6	PS Adresse+1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1
7	Direktwert OUT	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
8	Datenadresse an Bus	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
9	Register IN	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	Datenspeicher IN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
11	Datenspeicher OUT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
12	Temp1 ALU IN	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
13	Temp2 ALU IN	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
14	ALU OUT	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
15	Akku IN	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	Akku OUT	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Assemblerbefehl	MOV R1,A	ADD A,#const8	ORL dadr,#const8
Maschinenbefehl	E9	24CC	43DDCC