# *Direct Memory Access* (DMA)

IT05/4

## Inhalt:


- 1 Begriffsklärung DMA
  - DMA
  - UDMA
- 2 Evolution
  - Vorgänger
- 3 Funktionsweise
  - DMA Controller
  - Bus Arbitration
  - DMA Transferarten
- 4 Grafiken
  - DMA & UDMA Taktsignal
  - DMA Controller i8237
  - Zentraler DMA Controller
- 5 Quellen


Marko Oette

Digital unterschrieben von Marko Oette
DN: cn=Marko Oette, c=DE, o=-, ou=-, email=marko@oette.info
Grund: Ich bin der Verfasser dieses Dokuments
Datum: 2005.12.06 15:31:02 +01'00'

# 1. Begriffsklärung

### 1.1. DMA

- DMA = Direct Memory Access (*dt: Direkter Speicher Zugriff*)
- Internes Datentransferverfahren, welches es Komponenten wie Festplatten, Grafikkarten oder Soundkarten erlaubt direkt auf den Arbeitsspeicher zuzugreifen
- Transfer zwischen Speicher ←→ Peripherie; Peripherie ←→ Peripherie; CPU ←→ Speicher (Speicher ←→ Speicher) unter Umgehung des Prozessors, 1x in jedem Zyklus
- Erstmals mit dem IBM PC/AT eingeführt (Intel 8237 DMA Controller s. B1)

### 1.2. Ultra DMA

- Weiterentwicklung des DMA Modus von Intel und Quantum welches durch eine Datenübertragung im Burst Mode (s. 3.3) noch schnellere Transferraten erlaubt
- Transfer der Daten (16 Bit) 2 x pro Zyklus (*double transition clocking* s. B2*)*
- CRC *(cyclical redundancy checking)* hinzugefügt (Prüfung auf Fehler bei der Datenübermittlung)

# 2. Evolution

### 2.1. Vorgänger

- PIO = Programmed Input Output oder Programmable Input Output (*dt: Programmierte Ein-/Ausgabe*)
- Internes Hardwarekommunikationsprotokoll / -verfahren, welches es Komponenten wie Festplatten, Grafikkarten oder Soundkarten erlaubt mit Hilfe der CPU und durch Softwaresteuerung auf den Arbeitsspeicher zuzugreifen
- Hauptnachteil ist die u.U. enorme Auslastung des Prozessors und die langsamere Datentransferrate (RAM ←→ CPU → Register →...←→ Peripherie) (s. T1)

**Modi im Überblick:**

| Modus | Übertragungsrate | | Zyklus Dauer | |
|---|---|---|---|---|
| IDE (ATA) PIO 0 | 3,33 | MByte / sek. | - | |
| IDE (ATA) PIO 1 | 5,22 | MByte / sek. | - | |
| IDE (ATA) PIO 2 | 8,33 | MByte / sek. | - | |
| IDE Multiword-DMA 0 | 4,16 | MByte / sek. | 240 | ns |
| IDE Multiword-DMA 1 | 13,33 | MByte / sek. | 160 | ns |
| IDE Multiword-DMA 2 | 16,66 | MByte / sek. | 120 | ns |
| E-IDE (Fast ATA-2) PIO 3 | 11,11 | MByte / sek. | - | |
| E-IDE (Fast ATA-2) PIO 4 | 16,66 | MByte / sek. | - | |
| Ultra-DMA/0 (ATA/16) | 16,66 | MByte / sek. | 240 | ns |
| Ultra-DMA/1 (ATA/25) | 25,0 | MByte / sek. | 160 | ns |
| Ultra-DMA/2 (ATA/33) | 33,33 | MByte / sek. | 120 | ns |
| Ultra-DMA/3 (ATA/44) | 44,4 | MByte / sek. | 90 | ns |
| Ultra-DMA/4 (Ultra-ATA/66) | 66,66 | MByte / sek. | 60 | ns |
| Ultra-DMA/5 (Ultra-ATA/100) | 99,99 | MByte / sek. | 40 | ns |
| Ultra-DMA/6 (Ultra-ATA/133) | 133 | MByte / sek. | (30 | ns) |

T1: Transfer Modi im Überblick

# 3. Funktionsweise

## 3.1. DMA Controller

- Ein DMA Controller ist generell jede Komponente, welche auf den Arbeitsspeicher zugreifen kann (lesen oder schreiben) ohne dafür den Prozessor zu beanspruchen
- Es gibt verschiedene Controller:
  - Zentralisierte DMA Controller:
    - Spezieller DMA Controller (im Bussystem siehe B3)
    - Die Komponente die Daten lesen oder schreiben möchte, sendet eine Anfrage an den zentralen DMA Controller
    - Dieser arbeitet diese ab, und benachrichtigt die Komponente sobald der Vorgang abgeschlossen ist
  - Dezentrale DMA Controller:
    - Jede Komponente die direkt auf den Arbeitsspeicher zugreifen möchte, reserviert sich Teile des System Busses beim Arbiter (siehe 3.2), adressiert selbstständig die Arbeitsspeicherbereiche auf die Sie zugreifen möchte, überträgt die Daten mittels des Zentralen DMA Controllers und löst einen Interrupt aus
    - Diese Komponenten werden wenn sie die Kontrolle über den Systembus haben als „Busmaster“ bezeichnet
- DMA Controller bieten den Vorteil, dass der Prozessor, während der DMA Controller die Daten überträgt bereits andere Aufgaben ausführen kann, sofern die dafür notwendigen Ressourcen nicht durch den DMA Controller belegt sind, oder gemeinsam genutzt werden (*Time Sharing*)
- 100 Prozentig wird der Prozessor jedoch nicht vom DMA Vorgang ausgeschlossen, er ist z.B. immer noch für die Berechnung von Adressen und für die Gerätesteuerung zuständig
- Spezielle I/O Prozessoren können diese Aufgaben erledigen, sofern sie vorhanden sind

## 3.2. Bus Arbitration

- Unter Arbitration (*dt: Schlichtung*) versteht man das Aushandeln des Busmasters zwischen allen Komponenten die DMA Zugriff fordern, sprich Busmaster werden wollen
- Der Bus Arbiter (*dt: Schlichter, oder besser Schiedsrichter*) bekommt von jeder Komponente die DMA Zugriff fordert ein HRQ (*Hold Request*) Signal und fügt diese in eine Warteschlange ein
- Der Arbiter wartet anschließend auf das EOP (*End Of Process*) Signal des aktuellen Bus Masters und sendet dann ein HLDA (*Hold Acknowledge*) Signal an die nächste wartende Komponente
- Möchte eine wartende Komponente wissen, wann sie an der Reihe ist, so kann sie ein BGA (*Bus Grand Acknowledge*) Signal an den Arbiter senden, und erhält daraufhin ihre Position in der Warteschlange (Dieses Signal ist nur bei Systemen mit Dezentralem DMA vorhanden)
- Der Bus Arbiter kann entweder des Prozessor selbst sein (Einfaches System mit einer CPU und höchstens einem DMA Controller) oder ein externer zentraler / externe dezentrale Arbiter (Komplexeres System)
- Sind mehrere externe Arbiter vorhanden sind diese untereinander verbunden, und handeln ihre Prioritäten aus
- Ein DMA Controller kann auch gleichzeitig ein Arbiter sein

### 3.3. DMA Transferarten

- Single Mode (*Cycle Stealing*)
  - Im Single Mode überträgt der DMA Controller bei jedem Taktzyklus („Durchlauf“) nur ein Byte, danach ist der Bus wieder freigegeben
  - Im aktuellen oder nächsten Taktzyklus wird wiederum eine Anfrage an den Arbiter gestellt und ggf. ein weiteres Byte übertragen
  - Dadurch wird dem Prozessor für kurze Zeit im Zyklus der Busmaster Status entzogen („gestohlen“)

- Multiword DMA ist eine Ableitung vom Single Mode DMA, bei welchem nicht nur ein Byte (*word*) pro Zyklus sondern mehrere, meist 2 übertragen werden

- Burst Mode (*Block Transfer Mode*)
  - Im Burst Mode wird der komplette Datenblock übertragen und nicht nur ein oder zwei Byte (abhängig vom Single- oder Multiword Transfermodus)
  - Erst wenn das Leseregister leer ist, oder der Controller das EOP Signal sendet, wird der Bus wieder freigegeben

- Demand Transfer Mode (Transfer auf Anforderung)
  - In diesem Modus muss an den DMA Controller eine Anforderung gestellt werden, und dieser wiederum arbeitet diese dann entweder im Single Mode oder Burst Mode ab

- Auto Initialize
  - In diesem Modus wird die DMA Operation (Single-Mode) nachdem sie beendet wurde wiederholt, solange bis das DREQ Signal aus ist, sprich das anfordernde Gerät / Programm keine Aktualisierung mehr benötigt. (Wird verwendet um z.B. Speicherblöcke zu aktualisieren)

### 3.5. Ablauf eines DMA Transfers

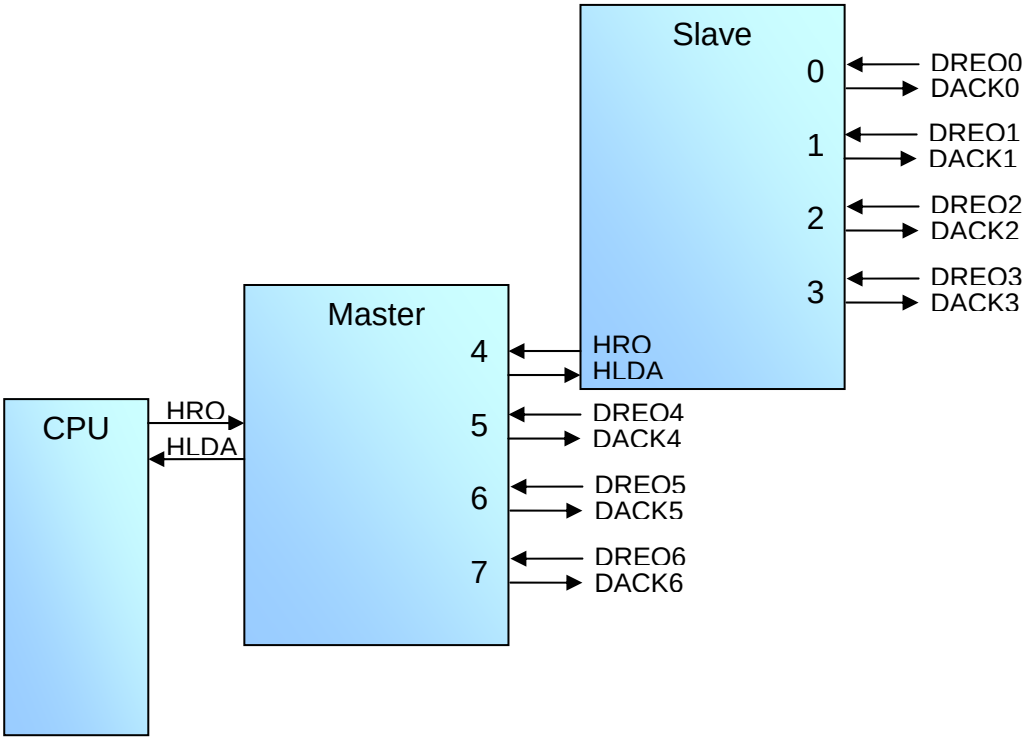
Ablauf anhand des Beispiels eines Diskettencontrollers:

1. Der Floppydisk Controller (FDC) liest ein Byte vom Datenträger und soll dies an der Position x00123456 speichern
2. Er legt deshalb am Kanal 2 des DMA Controllers das DREQ Signal an
3. Der DMA Controller stellt daraufhin fest ob der BUS frei ist, und ob andere Kanäle offen sind oder eine höhere Priorität haben
4. Der DMA Controller fordert den Prozessor auf den Bus Freizugeben indem er dem Prozessor das HRQ Signal sendet
5. Der Prozessor schließt daraufhin die aktuelle Operation ab und übergibt die Buskontrolle an den DMA Controller, indem er das HLDA Signal sendet
6. Abhängig vom Typ des Prozessors kann dieser nun noch einige Aufgaben für die nicht alle Teile des Bussystems benötigt werden ausführen
7. Der DMA Controller setzt nun die Adresse und sendet das DACK Signal über Kanal 2
8. Der FDC muss nun das Byte für diese Adresse übertragen
9. Nun wartet der DMA Controller einen Taktzyklus, und sofern das READY Signal vom FDC vorhanden ist (Transfer abgeschlossen) wird die Kontrolle an den Prozessor zurück gegeben (HRQ Signal wird entzogen)
10. Der Prozessor entzieht daraufhin das HLDA Signal und geht seiner Arbeit nach
11. Sind mehrere Byte zu kopieren wird dieser Vorgang wiederholt und am Ende das EOP Signal gesendet

Für einen Diskettensektor wird dieser Vorgang 512 Mal wiederholt. Danach (EOP + DACK2 Signal) muss der FDC einen Interrupt Signalisieren, damit die Arbeit fortgesetzt werden kann.

Wird im Block Mode auf Daten zugegriffen, wird im Schritt 9 die Kontrolle nicht an den Prozessor zurückgegeben, sondern gleich die Adresse inkrementiert und der Vorgang wiederholt, bis alle Bytes kopiert sind.

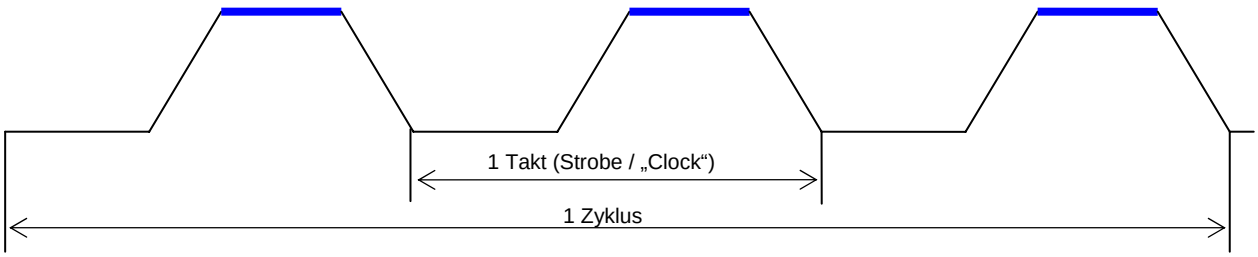
## 4. Grafiken



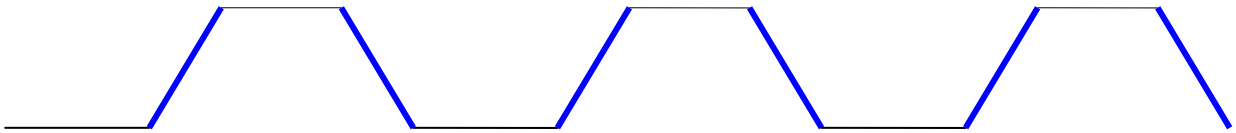

B1: Kaskadierung von zwei Intel 8237 DMA Controllern
Standartbelegung: Kanal 0: Speicherrefresh; 2: Diskettenlaufwerk; (3: Festplatte); 4: Kaskadierung mit Slave

© 2005 M. Oette

DMA Transfer: *single transition clocking*




UDMA Transfer: *double transition clocking*



B2: DMA und UDMA Taktsignale

= DMA Transfer

© 2005 M. Oette

**Bild ist aus rechtlichen Gründen nicht vorhanden. Siehe**
http://support.novell.com/techcenter/articles/img/ana1995050103.gif

B3: Zentraler DMA Controller (5) © Novell

## 5. Quellen

Texte:

- http://de.wikipedia.org/wiki/Programmed_Input_Output
- http://de.wikipedia.org/wiki/Ultra-DMA
- http://de.wikipedia.org/wiki/Direct_Memory_Access
- http://de.wikipedia.org/wiki/Busmaster
- http://www.bjoern-koester.de/iogrundlagen
- http://www.freebsd.org/doc/en_US.ISO8859-1/books/developers-handbook/dma.html
- http://www.webopedia.com/TERM/U/Ultra_DMA.html
- http://www.pcguide.com/ref/hdd/if/ide/modesUDMA-c.html
- http://ipr.ira.uka.de/perso/brinks/ps/mrt2vorlesKap2-1.pdf

Bilder:

- http://support.novell.com/techcenter/articles/img/ana1995050103.gif