Bussar och I/O

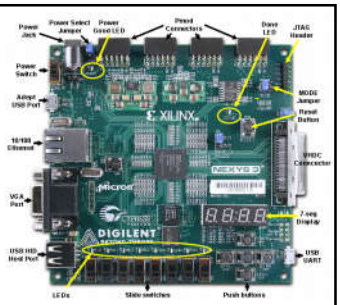
samt

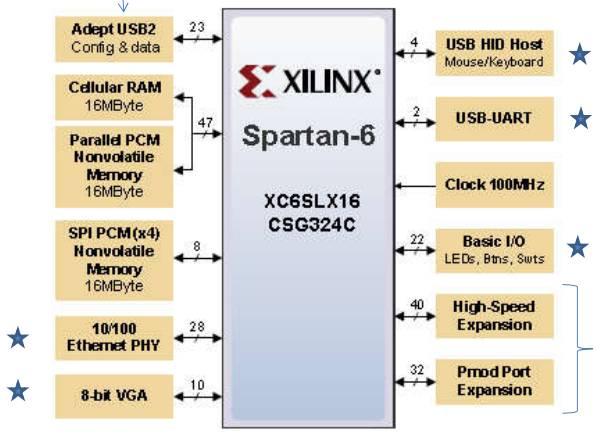
vad kan man göra med NEXYS3

1

Nexys3

Prog + spänningsmatning mha USB

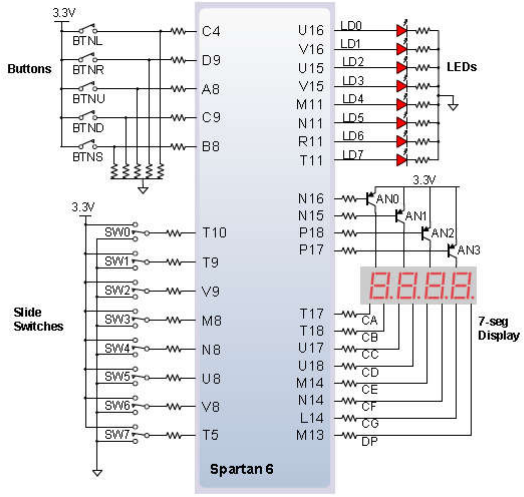




Expansions-kort ★

2

Basic I/O



3

Master.ucf (User Constraints File)

```
## 7 segment display
Net "seg<0>" LOC = T17 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L51P_M1DQ12, Sch name = CA
Net "seg<1>" LOC = T18 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L51N_M1DQ13, Sch name = CB
Net "seg<2>" LOC = U17 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L52P_M1DQ14, Sch name = CC
Net "seg<3>" LOC = U18 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L52N_M1DQ15, Sch name = CD
Net "seg<4>" LOC = M14 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L53P, Sch name = CE
Net "seg<5>" LOC = N14 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L53N_VREF, Sch name = CF
Net "seg<6>" LOC = L14 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L61P, Sch name = CG
Net "seg<7>" LOC = M13 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L61N, Sch name = DP

Net "an<0>" LOC = N16 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L50N_M1UDQSN, Sch name = AN0
Net "an<1>" LOC = N15 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L50P_M1UDQS, Sch name = AN1
Net "an<2>" LOC = P18 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L49N_M1DQ11, Sch name = AN2
Net "an<3>" LOC = P17 | IOSTANDARD = LVCMOS33; #Bank = 1, pin name = IO_L49P_M1DQ10, Sch name = AN3
```

I VHDL-koden

```
seg <= ”10110000”;     -- siffran 3
an <= ”1110”;          -- längst till vänster
```

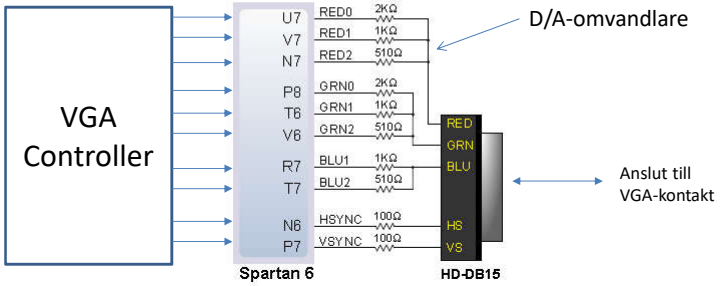
4

VGA port

I manualen finns siffror för 640*480-bild

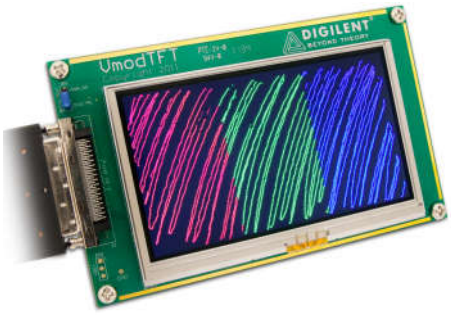
Komplexitet:

- Två räknare + lite avkodning s k VGA-motor
- …
- Avancerad GPU som kan programmeras



6

En Vmod



- LCD 480*272
- Resistive touchscreen
- Gränsnitt: se datablad!
- Styrs på ung. samma sätt som en bildskärm, dvs med synksignaler

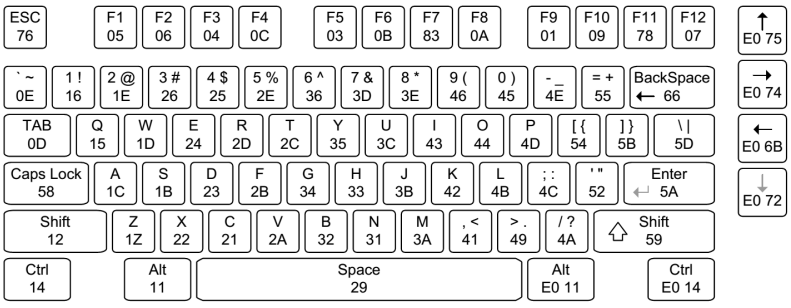
7

PC-Tangentbord

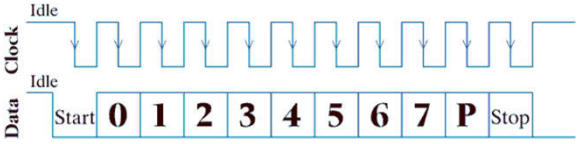
- Ansluts via USB
- Kommunicerar med PS/2
- Tangentkombinationer kan användas
- Avkodas med tillståndsmaskin

8

PC-Tangentbord



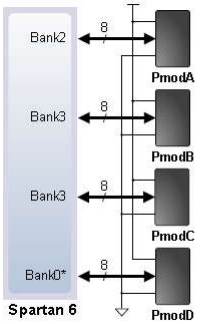
PS/2 Keyboard Scan Codes

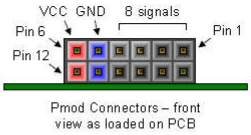


Tryck A ➔ 1C (make)
Tryck R ➔ 2D (make)
Släpp R ➔ F0,2D (break)
Släpp A ➔ F0,1C (break)

9

Pmod





Pmod Connectors – front view as loaded on PCB

Pmod Pinouts

JA1: T12	JB1: K2	JC1: H3	JD1: G11
JA2: V12	JB2: K1	JC2: L7	JD2: F10
JA3: N10	JB3: L4	JC3: K6	JD3: F11
JA4: P11	JB4: L3	JC4: G3	JD4: E11
JA7: M10	JB7: J3	JC7: G1	JD7: D12
JA8: N9	JB8: J1	JC8: J7	JD8: C12
JA9: U11	JB9: K3	JC9: J6	JD9: F12
JA10: V11	JB10: K5	JC10: F2	JD10: E12

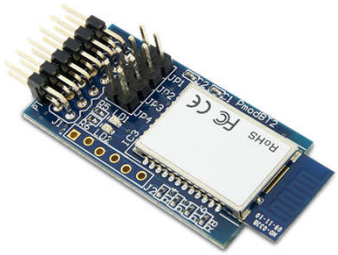
10

Några Pmods

HEX-tangentbord

- 16 tangenter, 0-F
- Saknar inbyggd kontroller, behöver ”scannas” drivrutin finns

BT2



- Blåtandsmodul
- UART-kommunikation

11

Några Pmods

OLED

- Monokrom
- Text / grafik 128*32
- Controller
- SPI

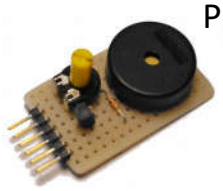
JSTK

- Two axis joystick
- Controller
- SPI

12

Några Pmods

Piezo



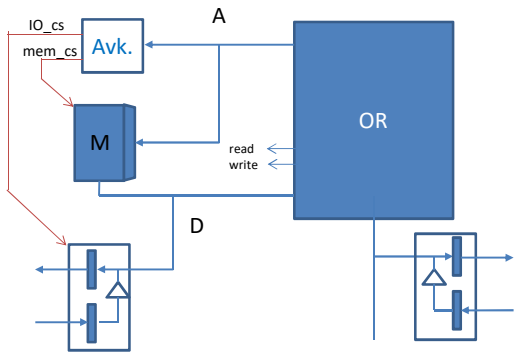
- Monohögtalare (Piezo)
- Bit banging

I²S

- Stereo”högtalare”
- I²S (seriellt)

13

Två sätt att koppla in I/O-kretsar

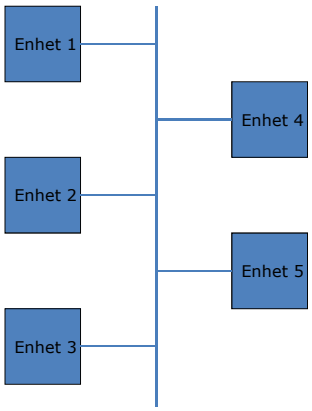


Memory-mapped I/O
- På den yttre databussen
- **I/O-registren som minnesceller**
- Vanliga instruktioner (LDA,STA)
- Adressavkodning måste fixas mm

I/O-mapped I/O
- Direkt på interna databussen
- **I/O-registren som interna register**
- Speciella instruktioner,
- Som kan vänta på att någonting hänt (fördelen med mikroprog. dator)

14

Parallella och seriella bussar



Buss = gemensamma ledningar för kommunikation mellan enheter. Endast en överföring åt gången.

Sändare/Mottagare = som det låter

Master/Slave = Master kan starta en överföring

Arbitrering = skiljedom, behövs för flera masters

On chip -> multiplexer
Off chip -> tristate

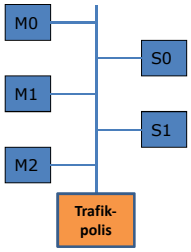
Dum <-> Smart

Seriell <-> parallell

Exempel på enheter:
- CPU
- Minne
- I/O – parallellport, UART, …
- I/O med DMA

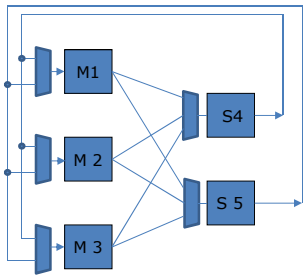
15

Några varianter

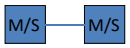


Multimaster-buss,
dubbelriktad databuss, tristate,
endast ett ”samtal” åt gången
(Jfr OR-datorn, mprog styr trafiken på bussen)

Korskoppling, crossbar
enkelriktade databussar



Punkt till punkt

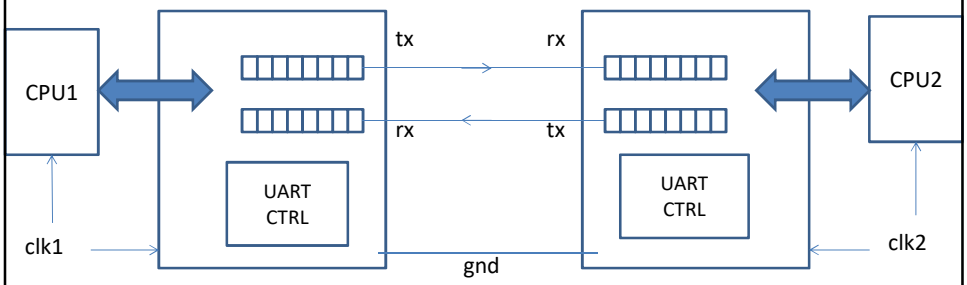


16

Kommunikation
mellan
datorer

RS232
UART

Universal Asynchronuous Receiver/Transmitter

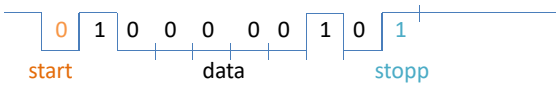


- Seriell
- Punkt till punkt
- Problem 1: UART1 och UART2 har inte samma klocka
  => synkronisering nödvändig
- Problem 2: hur vet CPU2 att ett tecken kommit in?
  hur vet CPU1 att ett tecken har sänts?
  => handskakning nödvändig

20

UART

Vår klocka = 100 MHz



Synkronisering: UART <-> UART

bithastighet: 115200 bit/s
1 bit = 868 CK, 1 tecken = 8680 CK, då CLK=100MHz
båda sidor har en räknare, som räknare 16 ggr / bit
S : skifta ut var 16:e CK
M: vänta på startbit, starta räknare,
skifta in mitt i bitarna

Handskakning: UART <-> CPU

flaggor (=bitar i kontrollregister)

rxfull = tecken har kommit in!	1: när tecknet kommit in
	0: när CPU:n läst tecknet
txempty = tecken har sänts!	1: när tecknet sänts
	0: när CPU:n skrivit tecken

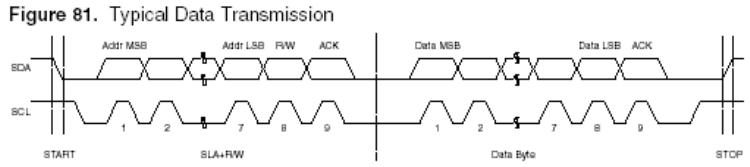
21

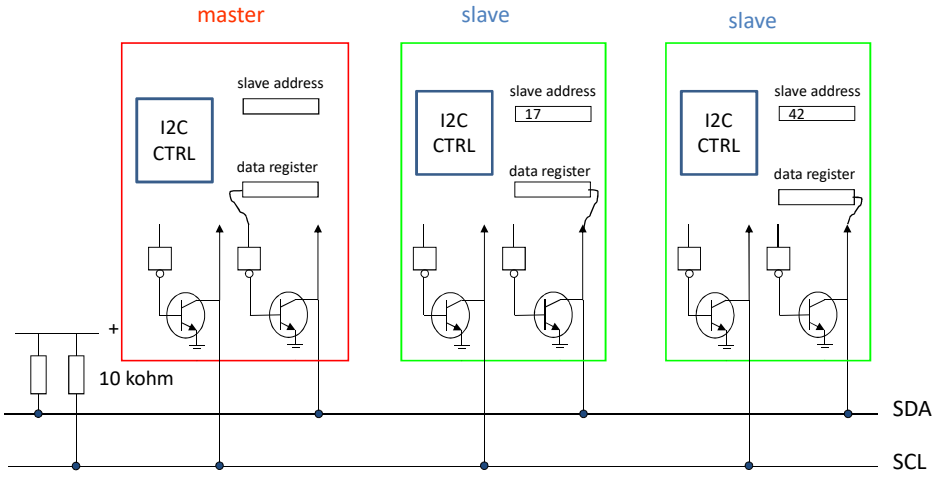
I^2C = Inter-integrated-circuit

- Seriell buss
- Philips
- Konsumentelektronik
- Kommunikation inom ett kretskort
- 2 trådar, ganska låg hastighet, billigt
- Finns i PC: läs av CPU temp, fläktvarvtal, minnestyp, …
- Upp till 127 enheter

22

I²C - Inter-Integrated-Circuit

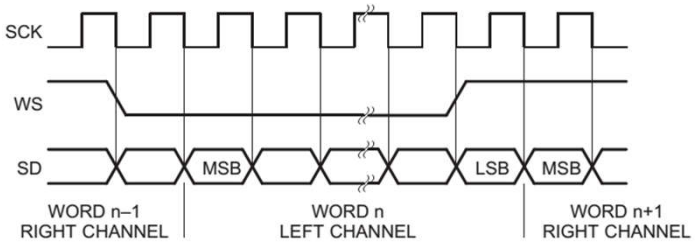




23

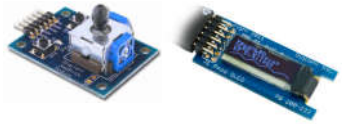
I²S Inter-IC Sound

- Seriell buss
- Philips
- Stereo , multiplexad höger/vänster-kanal
- Tvåkomplementskodat data med oberoende ordlängd
- T ex CD-ljud 44.1 kHz x 16 x 2 => 1.4112 MHz SCK



24

SPI



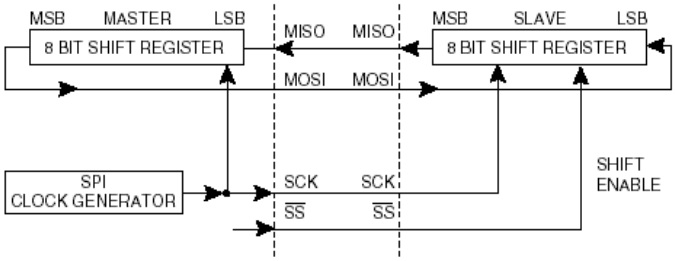
Serial Peripheral Interface – SPI

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the ATmega16 and peripheral devices or between several AVR devices. The ATmega16 SPI includes the following features:

- **Full-duplex, Three-wire Synchronous Data Transfer**
- **Master or Slave Operation**
- **LSB First or MSB First Data Transfer**
- **Seven Programmable Bit Rates**
- **End of Transmission Interrupt Flag**
- **Write Collision Flag Protection**
- **Wake-up from Idle Mode**
- **Double Speed (CK/2) Master SPI Mode**

f_{max} = 8/4 MHz

Figure 66. SPI Master-slave Interconnection

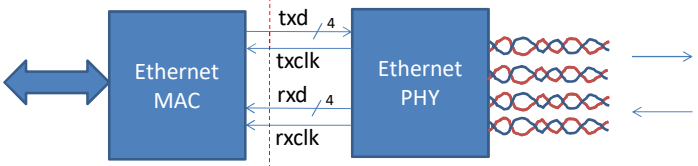


25

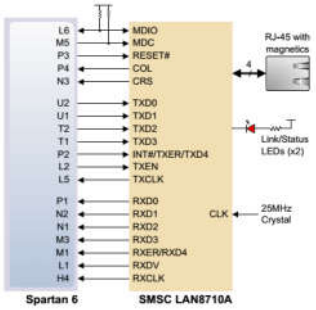
Ethernet PHY ?

Inne i FPGA:n

100 Mbit/s
seriellt



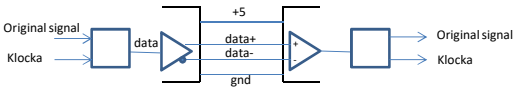
Media Access Control
”bygger paket av nibbles”
Nibble = halv byte, dvs 4 bitar



26

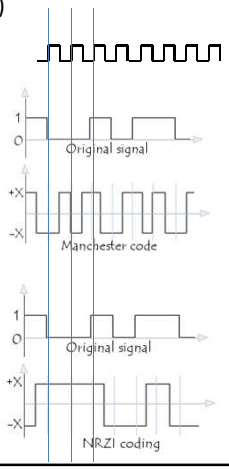
USB = Universal Serial Bus

1)



USB har bara dataledning

2)



0: upp
1: ner

0: behåll
1: ändra
bitstuffing

Low Speed – upp till 1,5 Mbit/s
Full Speed – upp till 12 Mbit/s
Hi-Speed – upp till 480 Mbit/s
Superspeed - upp till 5Gbit/s

3)

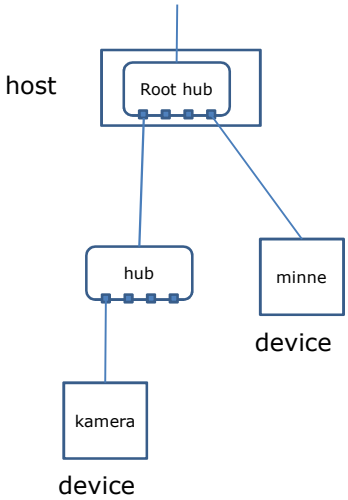
Kommunikationen är paketbaserad
Exempel:

Sync	Device ID	Data	CRC

Det finns många sorters paket!

27

USB = universal serial bus

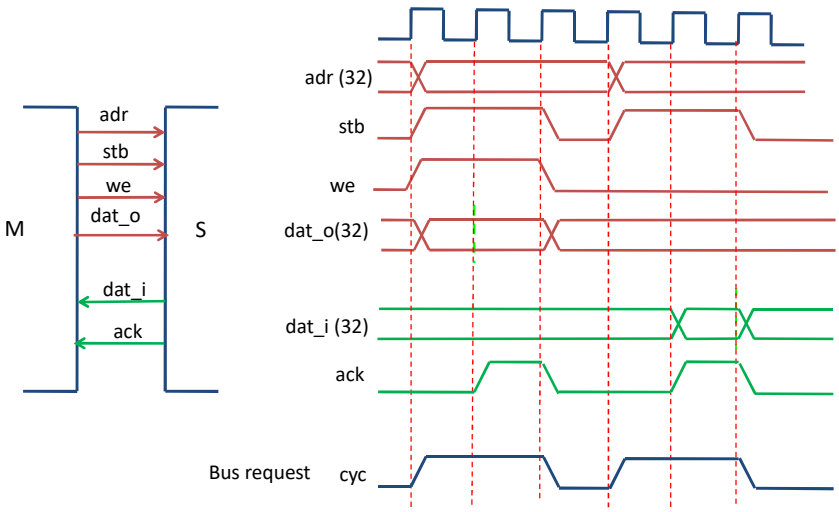


- 1 host
- upp till 127 devices
- Host kontrollerar all trafik
- Inga avbrott
- Each device sees all traffic from host
- A device does not see traffic from other devices
- A driver does not know topology of the network

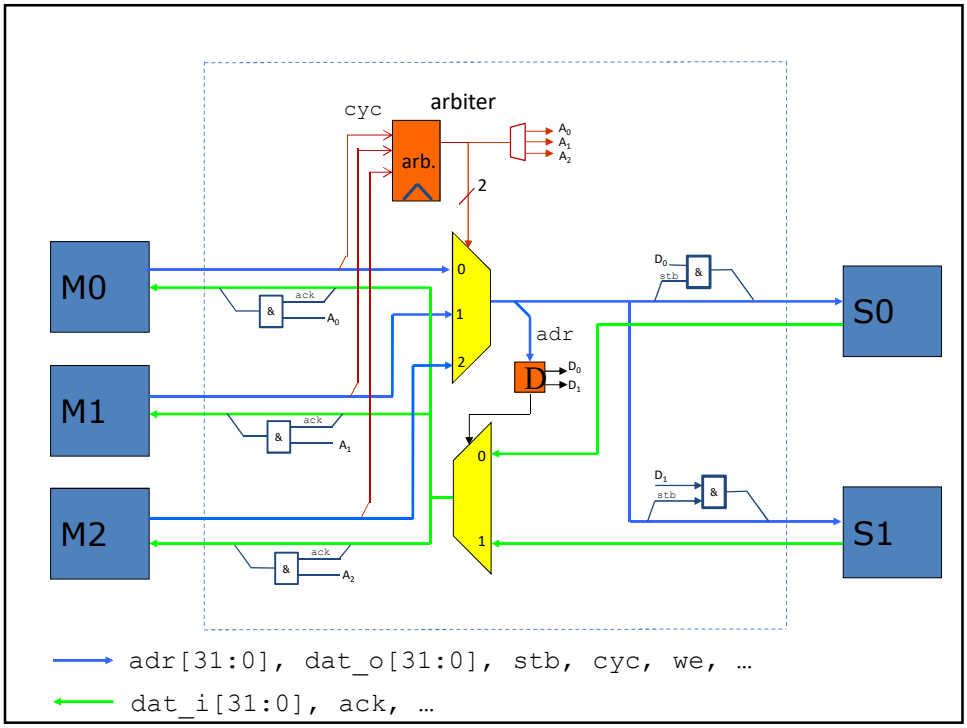
28

Enkel(?), parallell, asynkron buss

Wishbone: Open-Source-buss för FPGA, ASIC
on-chip bus, dubbla databussar



39



40

Sammanfattning

In-enheter

PC-tangentbord
PS/2

HEX-tangentbord
Scannas

Blåtandsmodul
UART

Joystick
SPI

Ut-enheter

VGA-monitor
Synksignaler

LCD med touch
Synksignaler

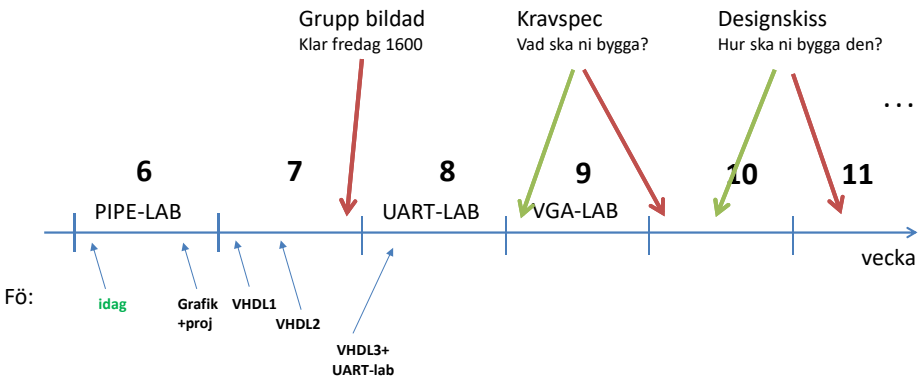
OLED-display
SPI

Monohögtalare
Bit banging

Stereo”högtalare”
I²S

45

Den närmaste framtiden



46