

* Synchronous & Asynchronous parallel communication

Synchronous : a clock at both transmitter & receiver is used to synchronize operations (impractical?)
Devices may be physically remote → distorted noisy clock

Asynchronous : Receiver & Transmitter use their own separate clocks to strobe or enable communication operations
i.e. "I've sent data" ↔ "I've received data"
messages to coordinate communication
"HANDSHAKING"

* simple strobed I/O is flexible for two reasons:

- ① It can provide two-way communications with a single read/write peripheral.
- ② It permits communications between MC6811 and two peripherals - one a read device second a write device.

STROBED I/O

is a technique to coordinate transfer of data in a simple way.

How is it used?

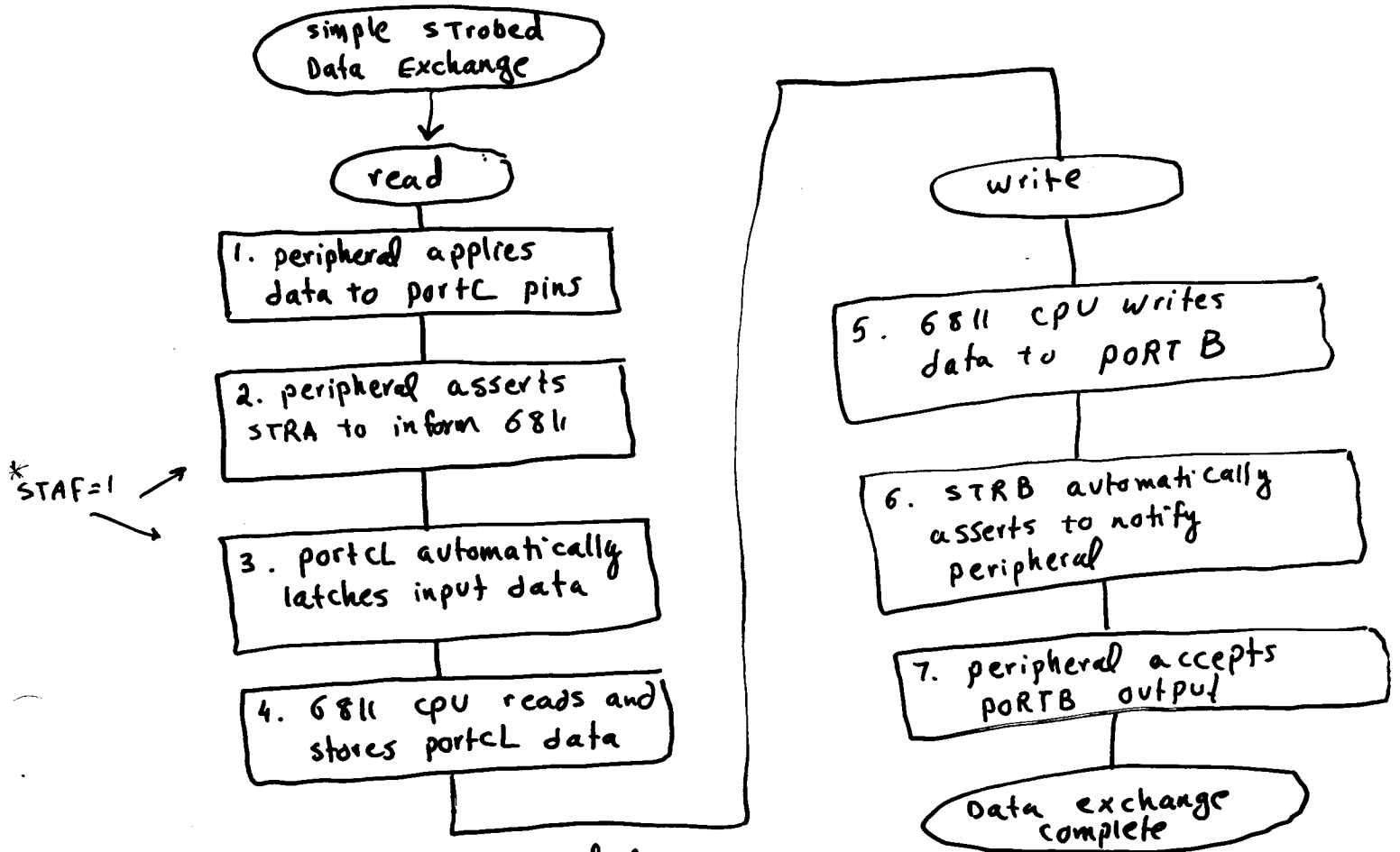
When MCU/MPU sends data to a peripheral, it tells the peripheral that data is available

(i.e) if peripheral uses flipflops to latch in data from MPU, it will need an extra strobe signal to clock or latch in data.

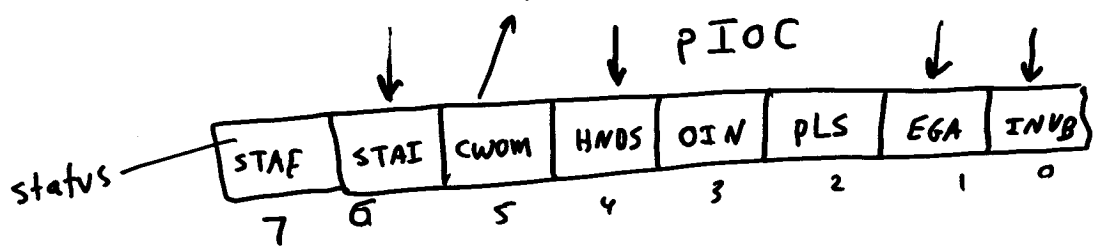
also If MCU is reading data from the peripheral, it has to know when the next data is sent \rightsquigarrow (So) peripheral can send a strobe signal that tells the MCU when new data is available

(or in other words) send a strobe signal to latch the data in a I/O port

SEQUENCE OF EVENTS IN A SIMPLE STROBED DATA TRANSFER



Normal or open drain



$\left\{ \begin{array}{l} \text{STAI} = 0 \\ \text{STAI} = 1 \end{array} \right. \begin{array}{l} \text{interrupt disabled} \\ \text{interrupt enabled} \end{array}$

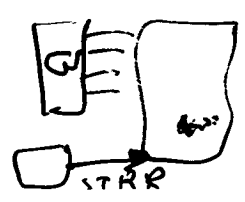
$\left\{ \begin{array}{l} \text{HNDS} = 0 \\ \text{HNDS} = 1 \end{array} \right. \begin{array}{l} \text{simple strobe mode} \\ \text{Full Handshake} \end{array}$

edge select $\left\{ \begin{array}{l} \text{EGA} = 0 \\ \text{EGA} = 1 \end{array} \right.$

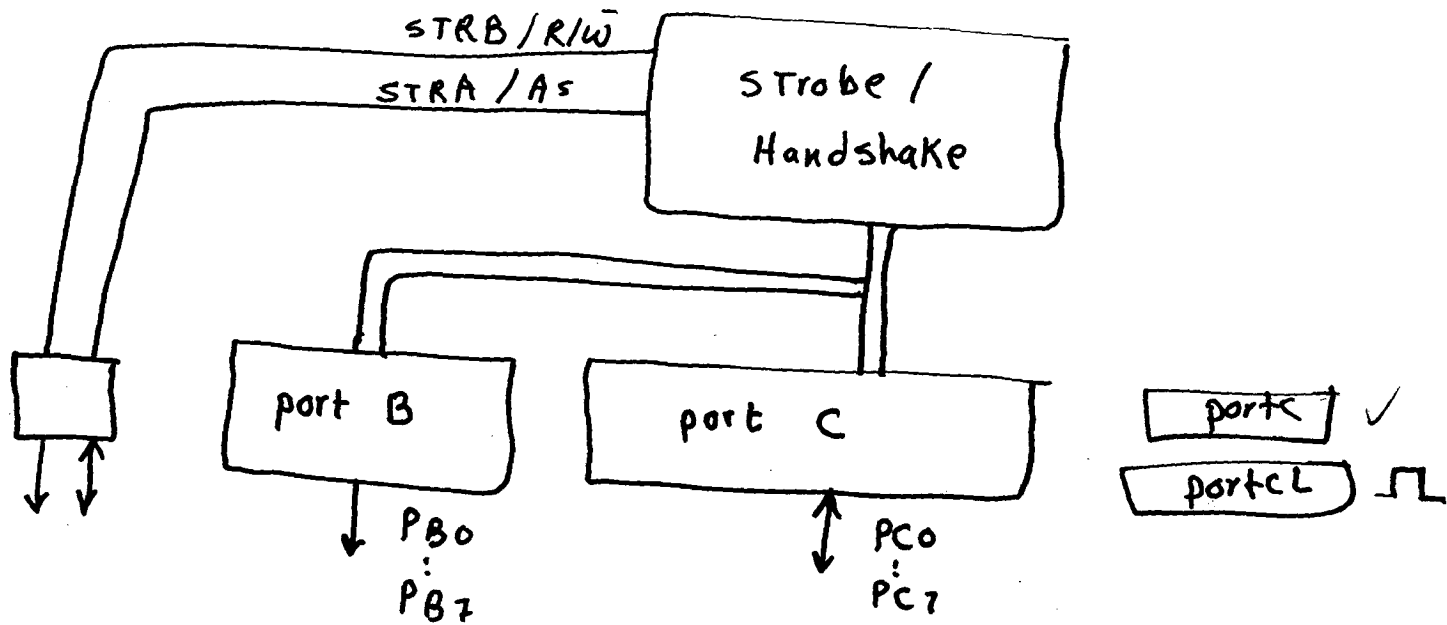
high to low transition asserts STRA
 low to high transition " "



$\left\{ \begin{array}{l} \text{INVB} = 0 \\ \text{INVB} = 1 \end{array} \right. \begin{array}{l} \rightarrow \text{STRB active low} \\ \rightarrow \text{STRB active high} \end{array}$



MC68HC11



The parallel I/O subsystem uses two control pins

- Strobe A (STRA) associated with port C
- Strobe B (STRB) associated with port B

Also

port C has two data registers

← accepts data no latching

- ① port C data register (\$1003) PORTC
- ② port C latched data Reg (\$1005) PORTCL

↓
is used for strobed and handshake I/O because it latches input data.

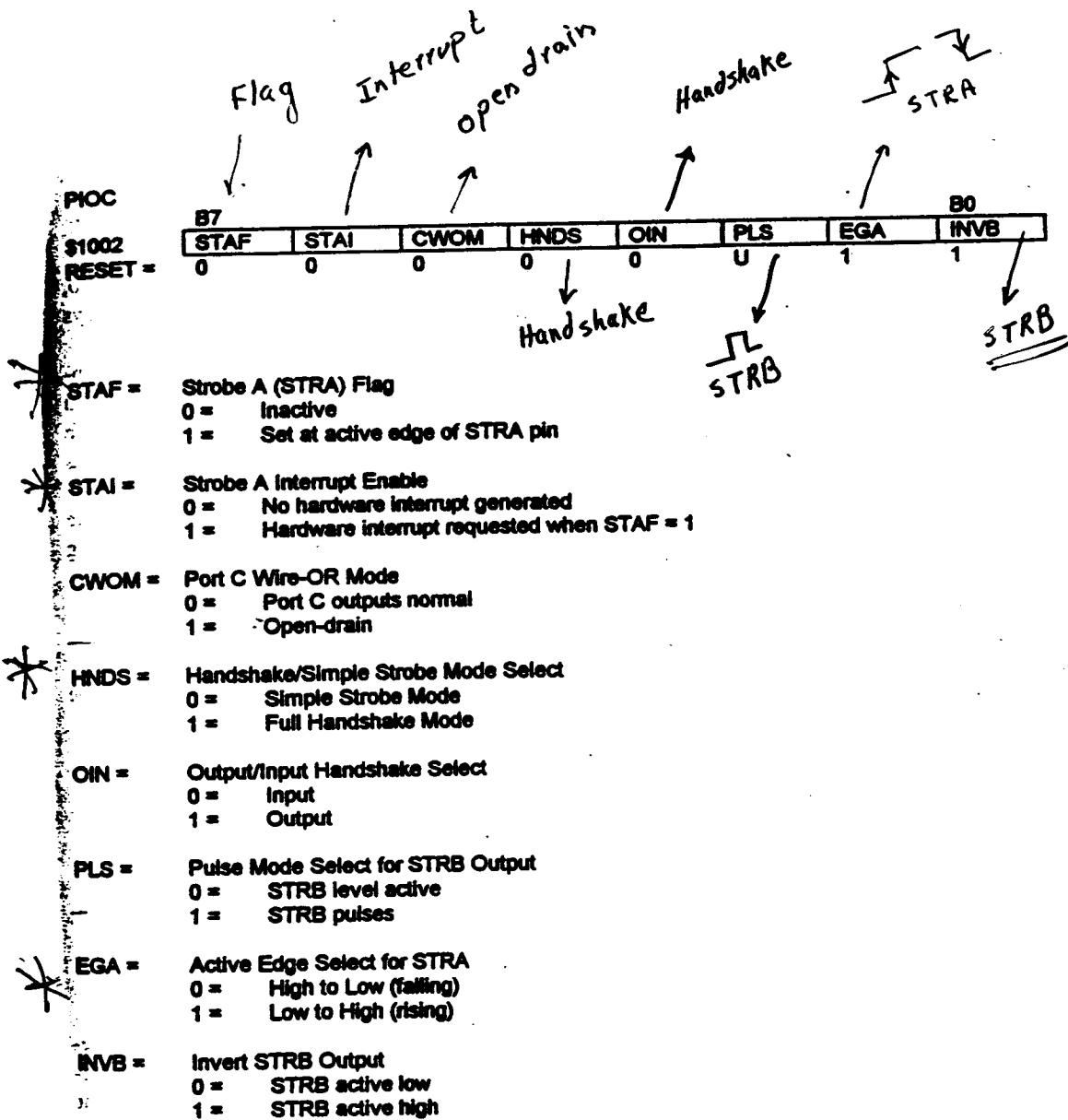
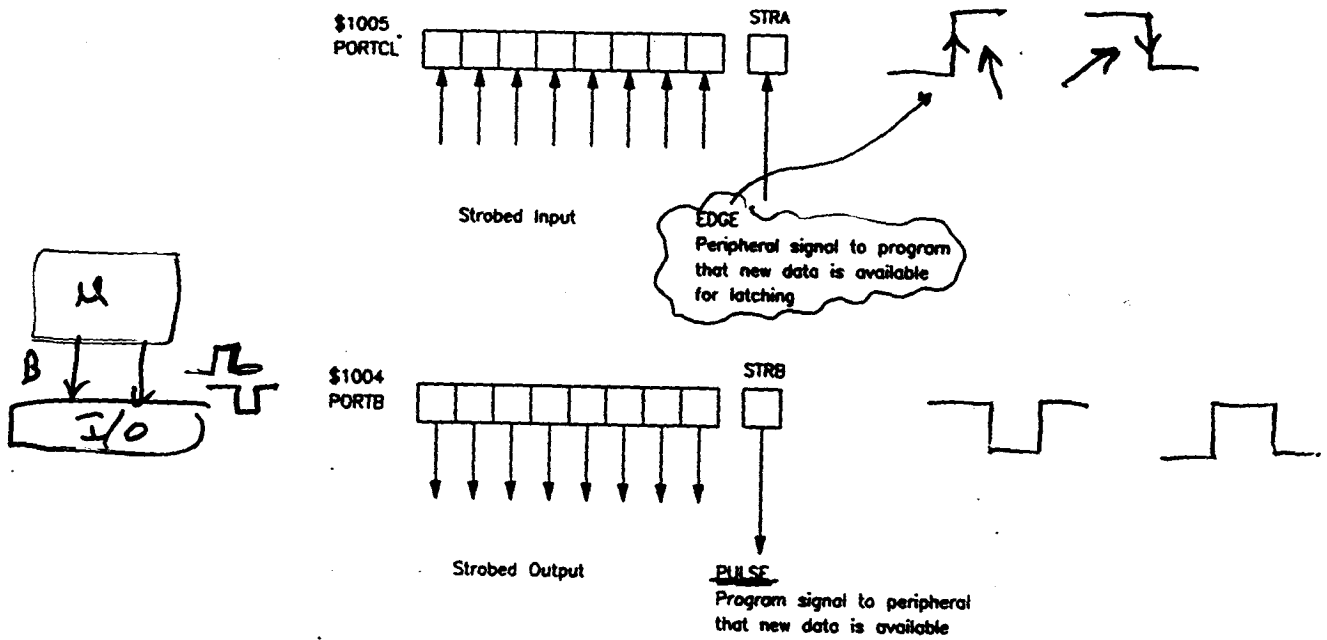


Figure 9.14 Parallel I/O Control Register

How to use the I/O subsystem
 for - strobe } synchronization
 - Handshake }



PIOC

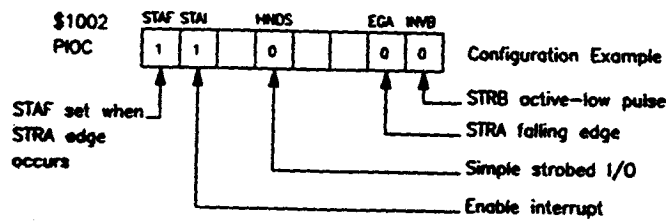


Figure 9.13 Strobed I/O

The parallel I/O control register pIOC (\$1002) is the control register for this system

↳ It determines how the MCU will behave for parallel I/O

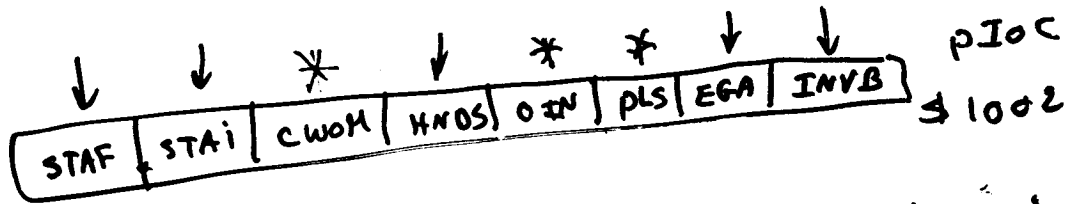
* we have to understand each bit within the pIOC

68HC11 strobed I/O

port B supports simple[^] strobed output

port C supports input as well as handshake Input/output

Controlled by P_{IOE} → parallel I/O control reg



* **STAF** : strobe flag - gets set when rising edge detected on STRA sig

* **STAI** : strobe interrupt enable. → hardware int

CWOM : wired or mode

0 → normal CMOS gate
1 → open drain output

* **HNOS** : 0 → simple strobe
1 → handshake mode

OIN : output/input handshaking

0 → input handshaking
1 → output handshaking

PLS : 0 → interlocked handshake
1 → pulse handshake

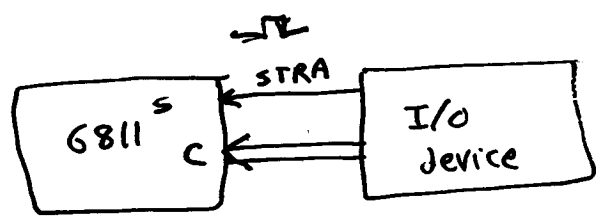
* **EGA** : Active edge for STRA

0 → falling edge
1 → rising edge

INVB :

0 : strB active level is logic 0
1 : " " " " = 1

Input Strobe



A peripheral has its output data lines connected to port C & the output control line (STROBE) connected to STROBE A (STRA)

(IMP) - if peripheral keeps pin STRA high, changes in port C lines, will also change the contents of data register **portC**

However, the changes will not show up in the data latch register **portCL**

- if STRA \downarrow falling edge, \rightarrow data on port C pins will be latched into **PORTCL**
 this causes **STAF** flag in PIOC


to set
 \rightarrow (IMP) if **STAI** bit was set by programmer an internal interrupt will occur.

\rightarrow (IMP) when MCU reads PIOC (STAF bit) and then reads data latch register **portCL** the (STAF bit) will CLEAR

REGBAS EQU \$1000 ;

PORTCL = \$05
 ODRC = \$07
 pIoc = \$2

*Listing 9.11
 *Strobed I/O Demo
 *Section of program to input 10 bytes
 *from port C and store them

When pin STRA
 sees , Flag
 STAF sets in
 pIoc

This is detected
 by

Flags then
 clears

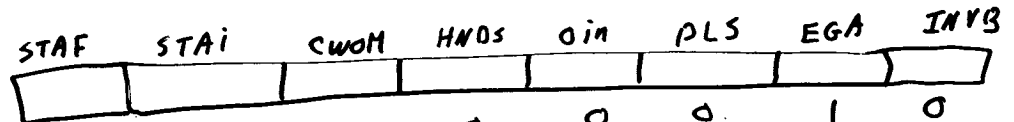
```

ORG    $100

LDX    #REGBAS
CLR    DDRC,X    ;configure port C as input
LDAA   #02       ;configure PIOC register
STAA  PIOC,X    ;active STRA is rising edge
LDY    #PTR      ;initialize storage pointer

*poll STAF for rising edge
CIN
BRCLR  PIOC,X,$80,CIN    ;mask
LDAA  PORTCL,X
      ;latch in port C input
      ;when STRA edge detected
      ;and store it
STAA  0,Y
INY
CPY   #PTR+10
BNE  CIN
STOP ;then stop

PTR EQU $B600
  
```



flag
 cleared

NO interrupt

simple
 strobe
 mode

Full Handshake input/output

Note

Strobed I/O can be used to transfer bytes of info between MCU & peripheral

But sometimes more control is required!
why?

① strobed input includes a signal from peripheral telling MCU that data is available **But** there is no signal from MCU to peripheral telling it that the MCU is ready to receive data

② same for strobed output.

protocols

When there is a transfer of data, there may have to be a set of rules to define when and how to transfer each Byte.

So → a protocol is a set of standard procedures used in data communication that coordinates the transmitting and receiving of information.

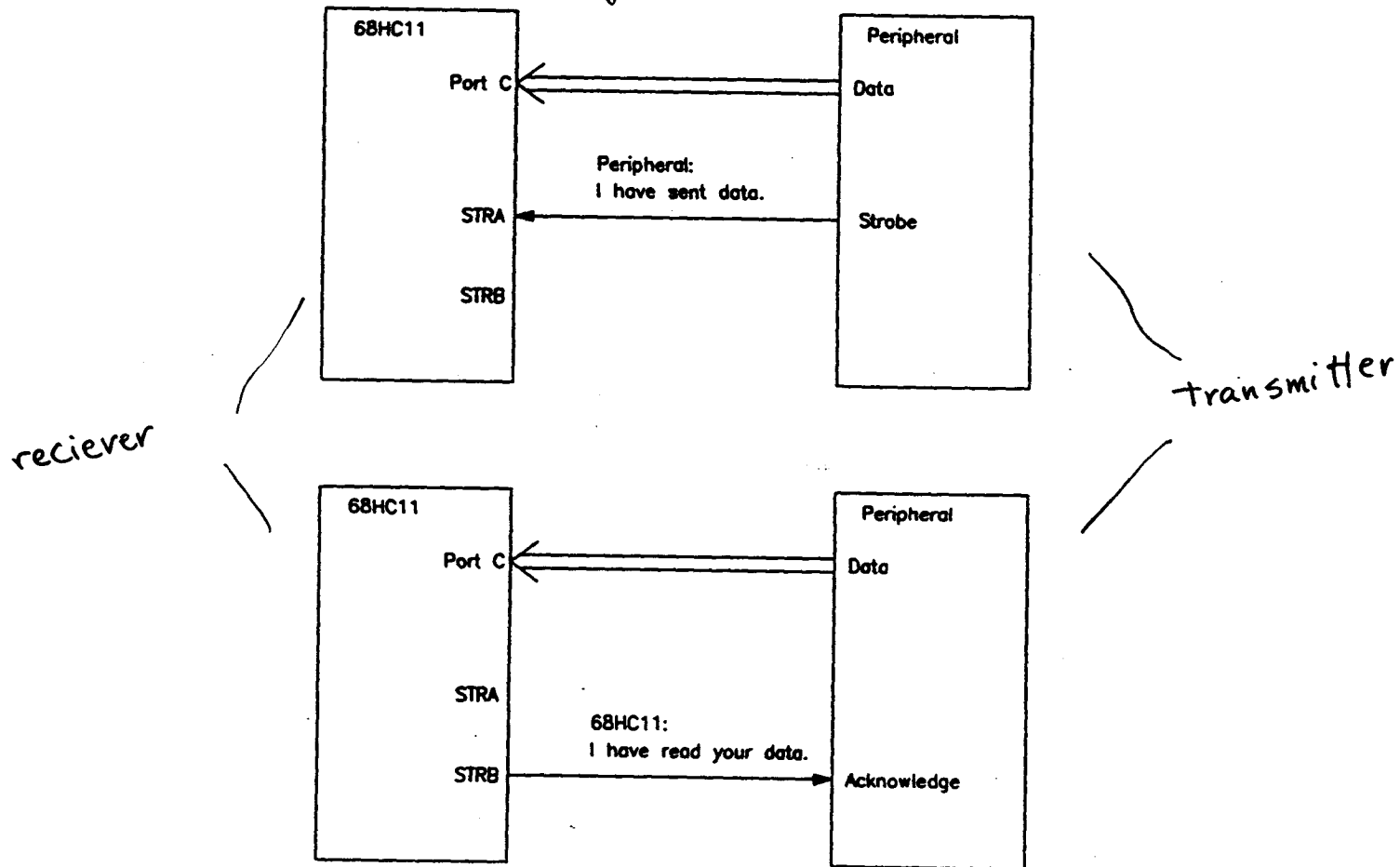


Figure 9.15 Input Handshaking

The handshake protocol is an agreement whereby the receiver acknowledges each unit of data it receives.

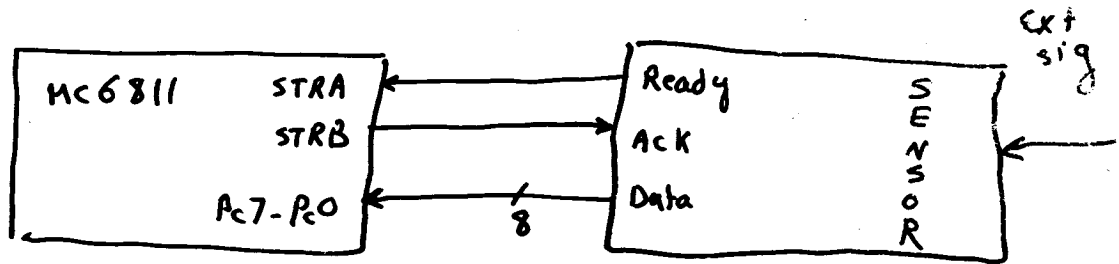
Transmitter waits for this ack before it sends the next unit.

68HC11 & Handshaking

The 68HC11 supports automatic handshaking for parallel I/O using port C

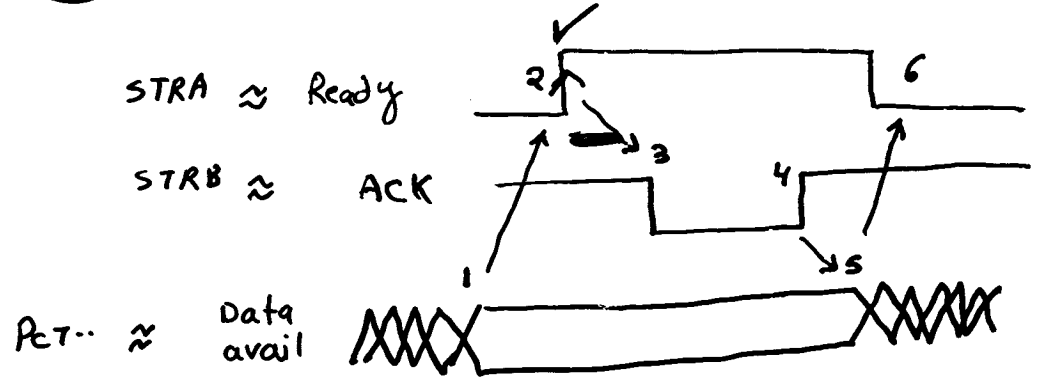
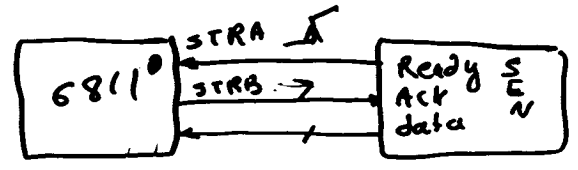
⇒ This is determined by configuring the control register PIOC

Example



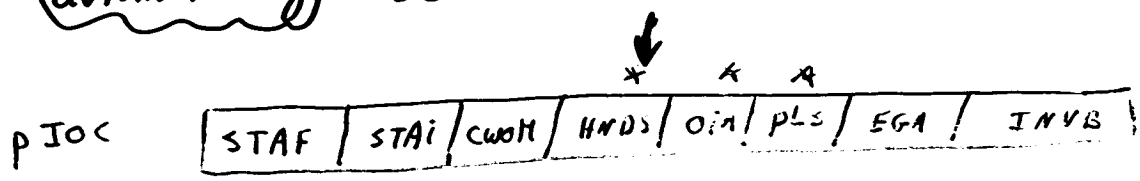
To input an 8-bit sensor reading, the software first waits for the next sensor reading to be ready.

Handshaking timing diagram For a sensor



Steps

- ① The input data from sensor is latched into portCL on active edge of STRA
- ② The rising edge of READY (STRA) also sets the flag "STAF"
- ③ 6811 uses gadfly synchronization to wait for 'STAF'
- ④ The two step sequence, read PIOC with STAF set followed by read portCL, will clear STAF flag.
- ⑤ with $PLS=0$, $INVB=1$ the rising edge of STRA automatically sets Ack (STRB) to 0



- PLS → pulse mode select for STRB output
 0 → level active ; 1 → STRB pulse

- 0 (1) HNDS → Handshake / simple strobe mode select
- 0 (1) OIN → output / input Handshake select

- ⑥ The ACK (STRB) output goes to 1 (inVB) when software read portCL
- ⑦ The falling edge of ACK is a signal from 6811 to the sensor signifying that the 6811 has begun to process the current input (i.e data is latched) BUT software did not yet read it

This interface is called Handshaked or Interlocked

because each event (1, 2, 3, 4, 5, 6) follows in sequence, one event after the other.

Imp ① Handshaking is a very reliable synchronization method when connecting devices from different manufacturers and at different speeds.

② It also allows you to upgrade one device (e.g, get a newer faster sensor) without redesigning both sides of the interface.

* Handshaking is used for SCSI & IEEE 488 BUS.

Centronics printer interface widely adopted printer interface protocol that uses pulse mode handshake protocol to achieve synchronization of data transfer.

two handshake line

- DATA STROBE

- ACKNLG

- data setup (50ns) → amount of time over which the data must be stable before the edge of CTRL signal that latches data

- data hold time (100ns) → the length of time over which data must remain stable after the edge of CTRL signal latches data arrives

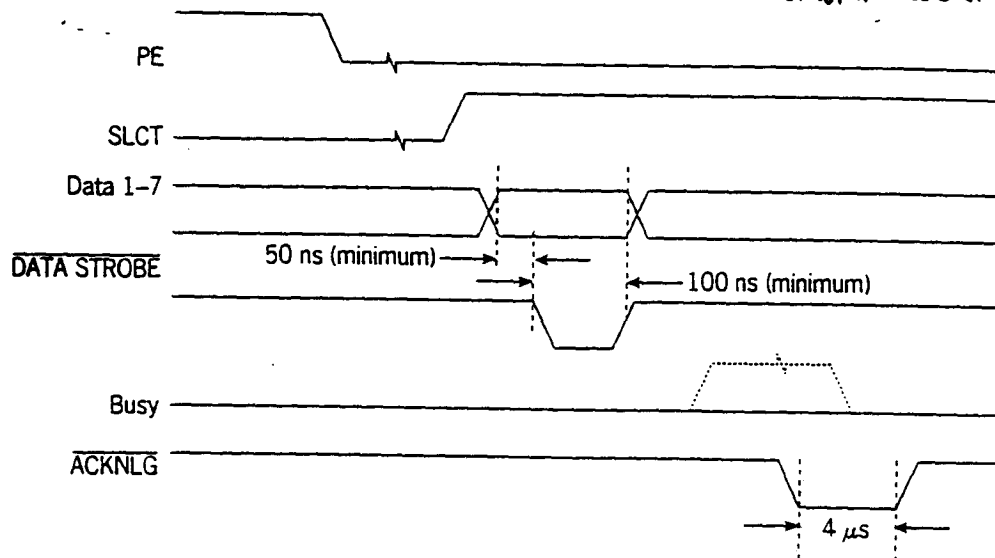


Figure 6.27 Centronics interface timing requirements

Centronics provides the following signals

- ① D7 - D1 data pins I/O
- ② Busy : output from printer (busy NOT ready to accept data because - offline - printing)
- ③ PE : output from printer (error)
- ④ SLCT : output from printer (printer on line)
- ⑤ DATA STROBE : input to printer (data is latched into printer by this signal)
- ⑥ ACKNLG : output from printer (ACK receiving info)

Design Example

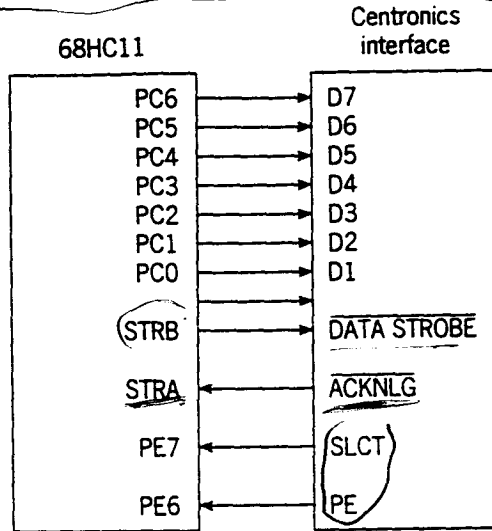
Interface a Centronics printer to port C

Designer expected to

- ① synchronize data transfer between port C & printer using pulse mode output handshake
- ② use polling to determine printer ready

③ write routines to

- initialize the printer
- output a character



STEPS

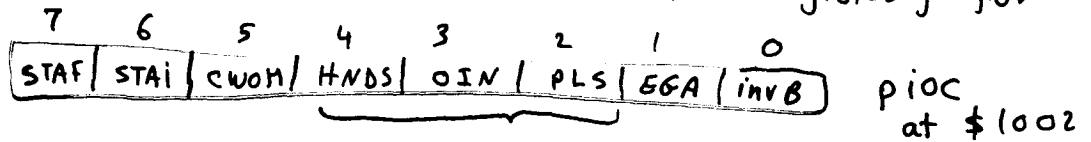
* possible connection

Figure 6.28 Interfacing a Centronics printer to port C

* verify that data setup & holdtime w.r.t Data Strobe satisfied

* set port C for output

* program PIOC (parallel I/O control register) for



- HNDS / OIN / PLS \equiv output pulse mode handshaking
- STAI \equiv strobe (A) interrupt disabled
- EGA \equiv strobe (A) falling edge as active edge
- CWOM \equiv normal port C pins
- INVB \equiv strobe (B) active low

Q modify routines for usage as interrupt driven output method