Arbeitskreis Hardware

Prof. Dr. Michael Rohs, Dipl.-Inform. Sven Kratz

michael.rohs@ifi.lmu.de

MHCI Lab, LMU München

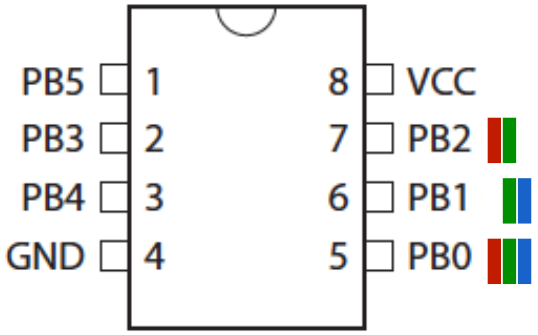
Schedule (preliminary)

Date	Topic (preliminary)
2.5.	Introduction to embedded interaction, microcontrollers, hardware & software tools
9.5.	*keine Veranstaltung (CHI)*
16.5.	ISP adapter soldering, AVR architecture
23.5.	LED displays, LED multiplexing, transistors, electronics basics
30.5.	AVR architecture, AVR assembler, sensors: light, force, acceleration, etc.
6.6.	PCB design & fabrication, EAGLE, 3D printing
13.6.	*keine Veranstaltung (Pfingsten)*
20.6.	Actuation: stepper motors, servo motors, I2C: interfacing to other chips
27.6.	USB to serial chips, storage on memory cards, capacitive sensors
4.7.	Displays (character LCDs, graphics LCDs), audio (speakers, amplification, op-amps)
11.7.	Communication: fixed-frequency RF, ZigBee, Bluetooth
18.7.	Project
25.7.	Project

INTERFACING HARDWARE

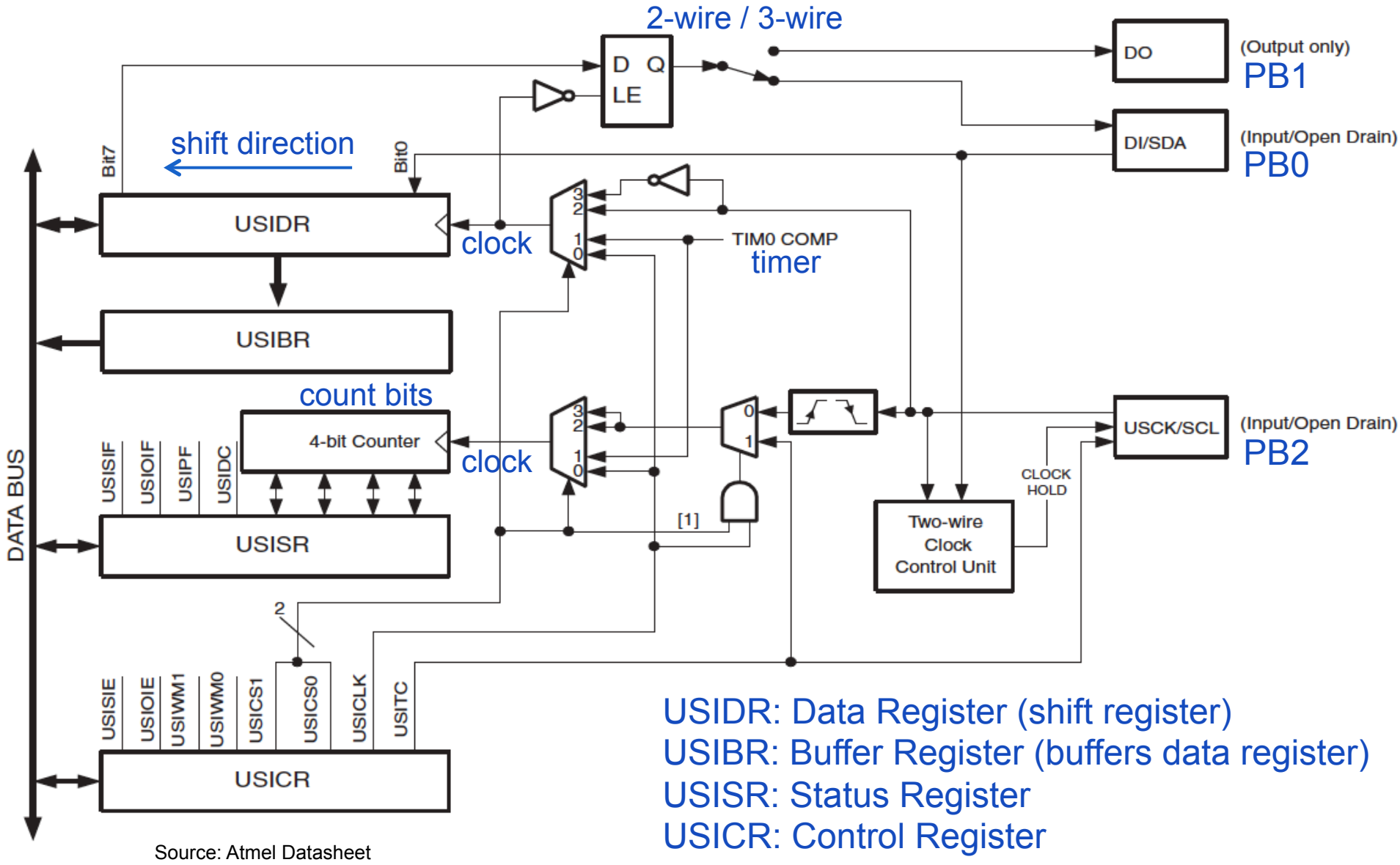
ATtiny45 Universal Serial Interface (USI)

- Two modes: two-wire mode, three-wire mode
- Serial Port Pins
- PB2
  - SCL: USI Clock (Two Wire Mode)
  - USCK: USI Clock (Three Wire Mode)
- PB1
  - DO: USI Data Output (Three Wire Mode)
  - MISO: SPI Master Data Input / Slave Data Output
- PB0
  - SDA: USI Data Input (Two Wire Mode)
  - DI: USI Data Input (Three Wire Mode)
  - MOSI: SPI Master Data Output / Slave Data Input



Source: Atmel Datasheet

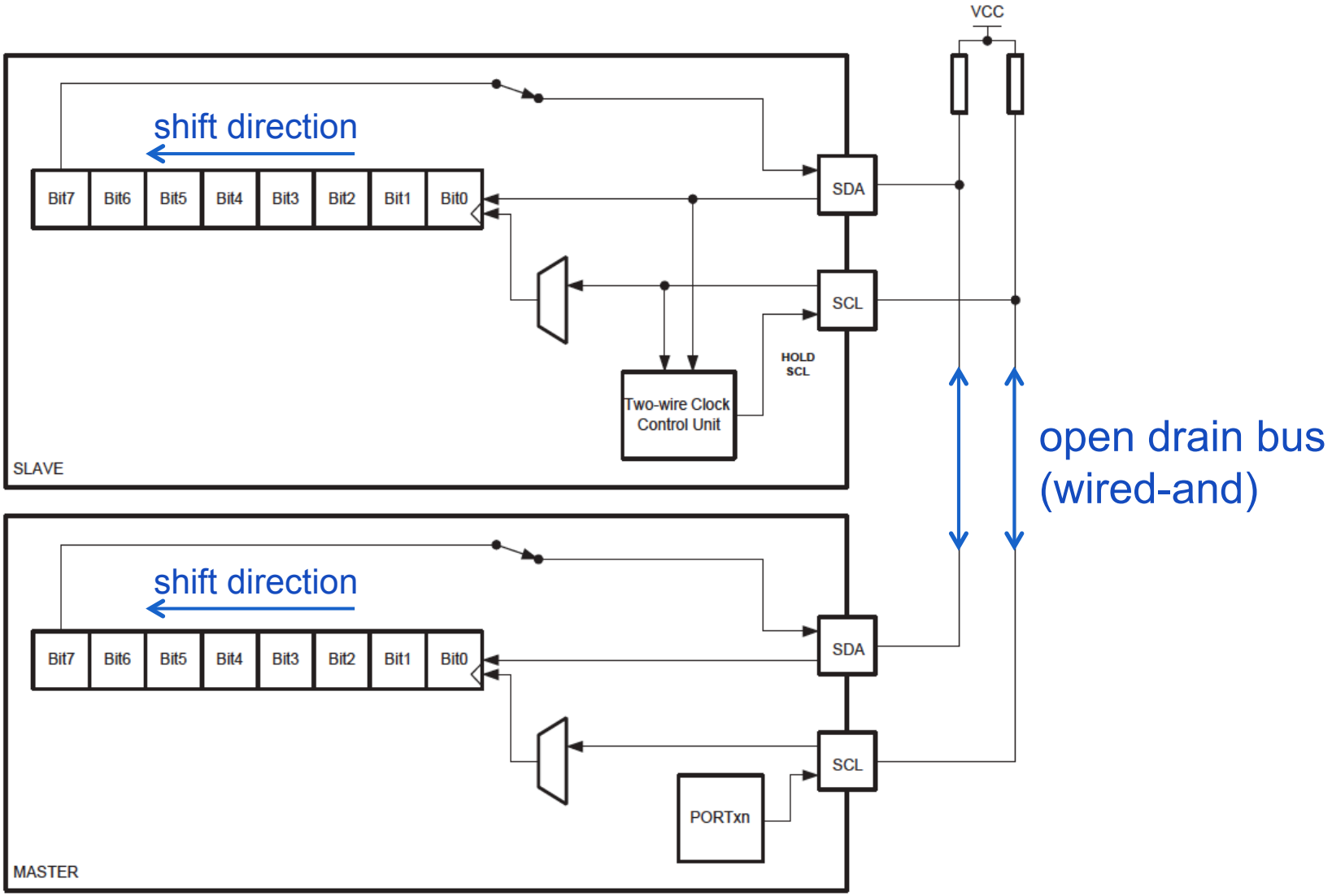
Universal Serial Interface, Block Diagram



USIDR: Data Register (shift register)
USIBR: Buffer Register (buffers data register)
USISR: Status Register
USICR: Control Register

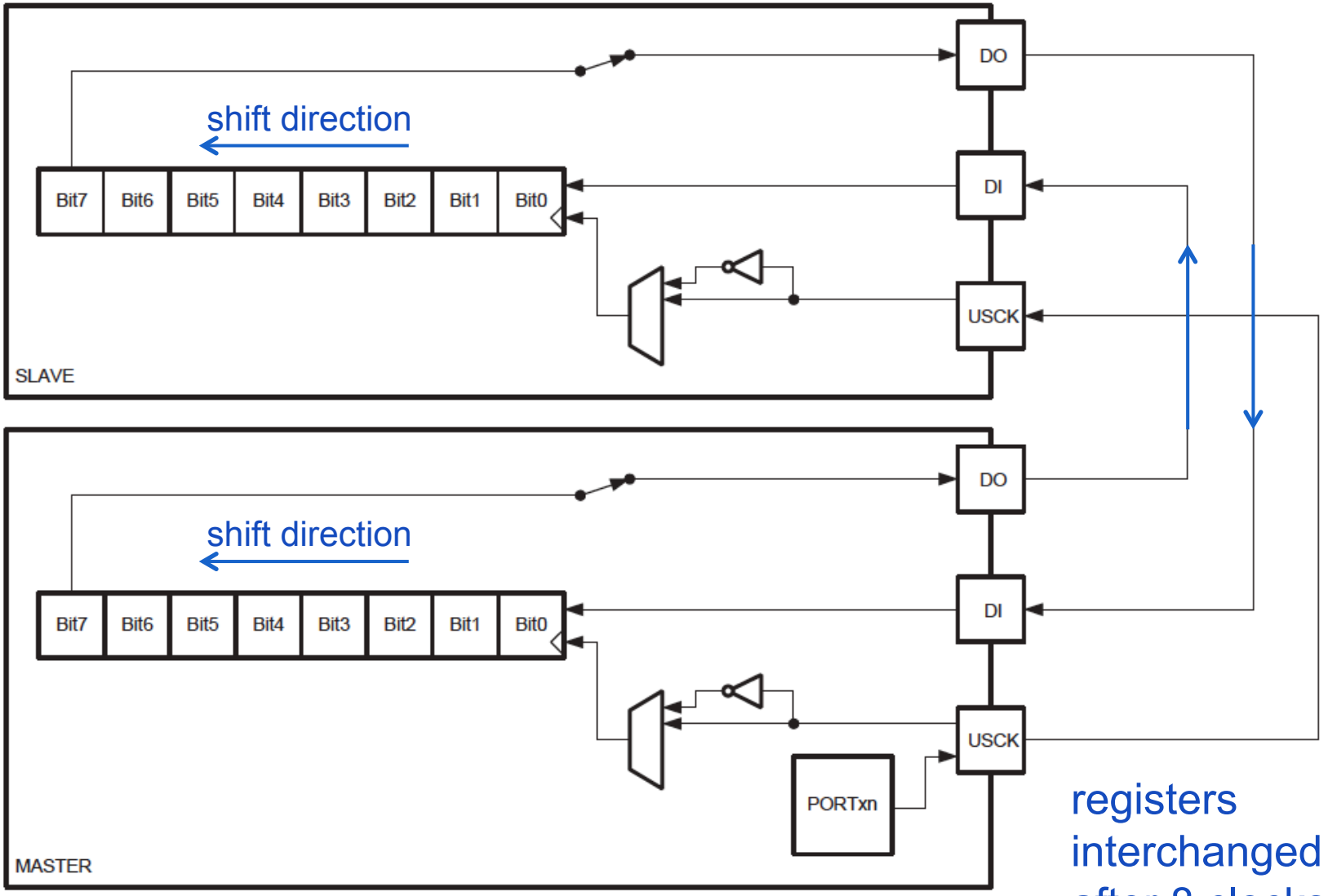
Source: Atmel Datasheet

USI Two-Wire Mode, I2C



Source: Atmel Datasheet

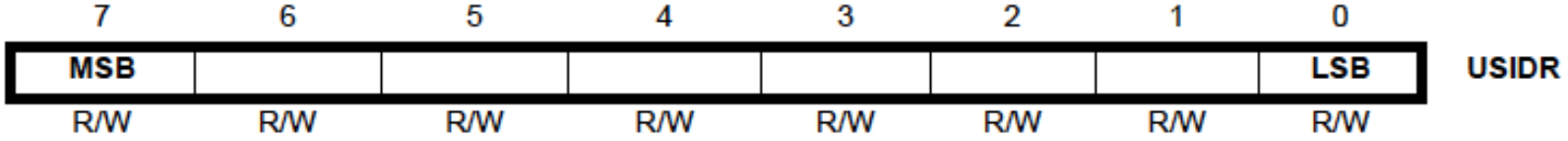
USI Three-Wire Mode



Source: Atmel Datasheet

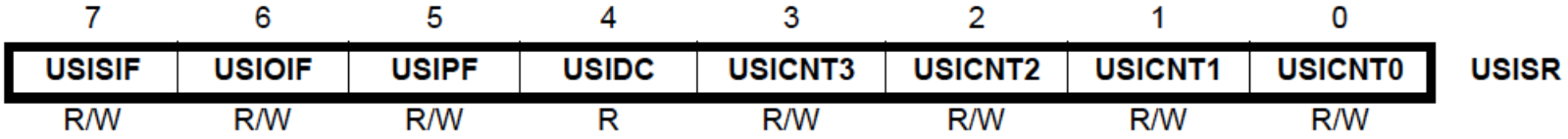
ATtiny45 USI Registers

- USIDR: Data Register (shift register)



- USIBR: Buffer Register (buffers data register)

- USISR: Status Register

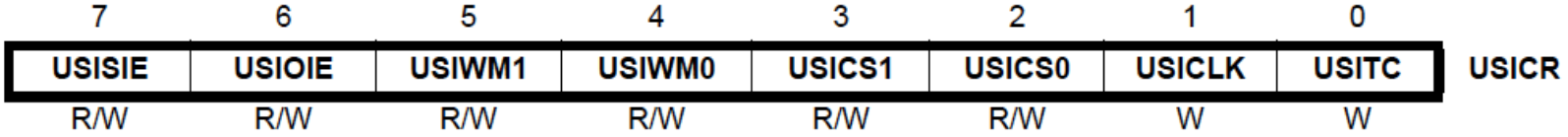


  - SIF: Start condition interrupt flag (two-wire mode)
  - OIF: Overflow interrupt flag (4-bit counter)
  - PF: Stop condition flag (two-wire mode, bus arbitration)
  - DC: data output collision (two-wire mode, bus arbitration)
  - CNT3..0: 4-bit counter (counts bits sent/received)

Figure sources: Atmel Datasheet

ATtiny45 USI Registers

- USICR: Control Register



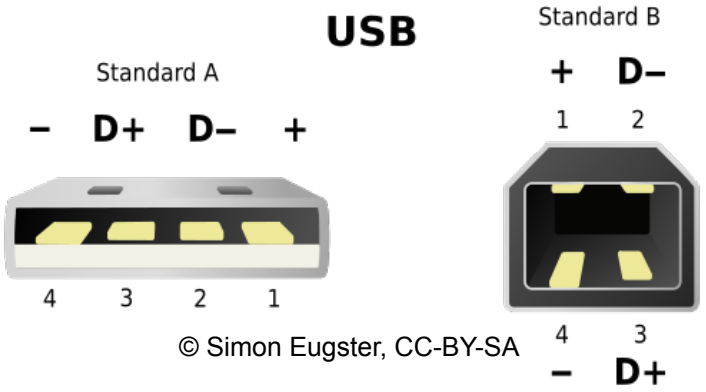
7	6	5	4	3	2	1	0	
USISIE	USIOIE	USIWM1	USIWM0	USICS1	USICS0	USICLK	USITC	USICR
R/W	R/W	R/W	R/W	R/W	R/W	W	W	

  - SIE: Start Condition Interrupt Enable
  - OIE: Counter Overflow Interrupt Enable
  - WM1,WM0: Wire Mode
    - 0,0: port pins operate as normal
    - 0,1: three-wire mode (DO, DI, USCK)
    - 1,0: two-wire mode (SDA, SCL, bi-directional, open drain)
    - 1,1: two-wire mode (as (1,1), but SCL held low on counter overflow)
    - data direction bits need to be set correctly
  - CS1,CS0: Clock Source Select
  - CLK: Clock Strobe
  - TC: Toggle Clock Port Pin

Figure sources: Atmel Datasheet

Universal Serial Bus (USB)

- High data rates (difficult to process with ATtiny)
  - USB 1.0: 1.5 Mbit/s (Low-Speed) and 12 Mbit/s (Full-Speed)
  - USB 2.0: 480 Mbit/s (High-Speed)
  - USB 3.0: 5 Gbit/s (Super-Speed)



Pin 1	V_{CC} (+5 V)
Pin 2	Data-
Pin 3	Data+
Pin 4	Ground

- 500mA max. (at V_{CC} = 5V)

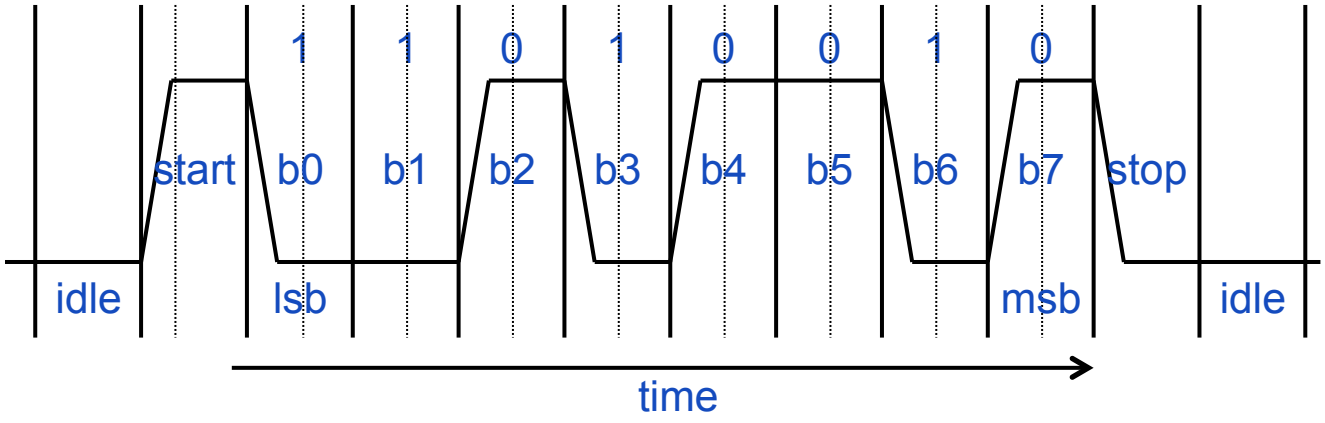
- Some AVRs have built-in USB modules

Universal Asynchronous Receiver/ Transmitter (UART)

- Sequential transmission/reception of a sequence of bits
- Framing (start bit = 0, stop bit = 1; 8 data bits, no parity bit)

Start (0)	Bit 0 (lsb)	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7 (msb)	Stop (1)

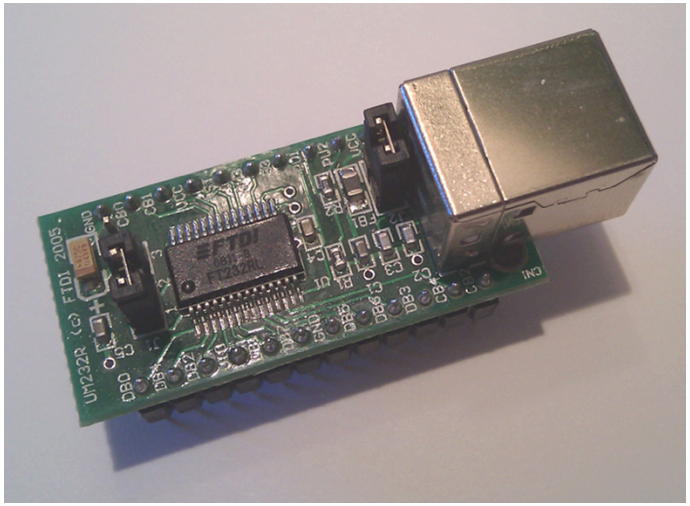
- Timing diagram

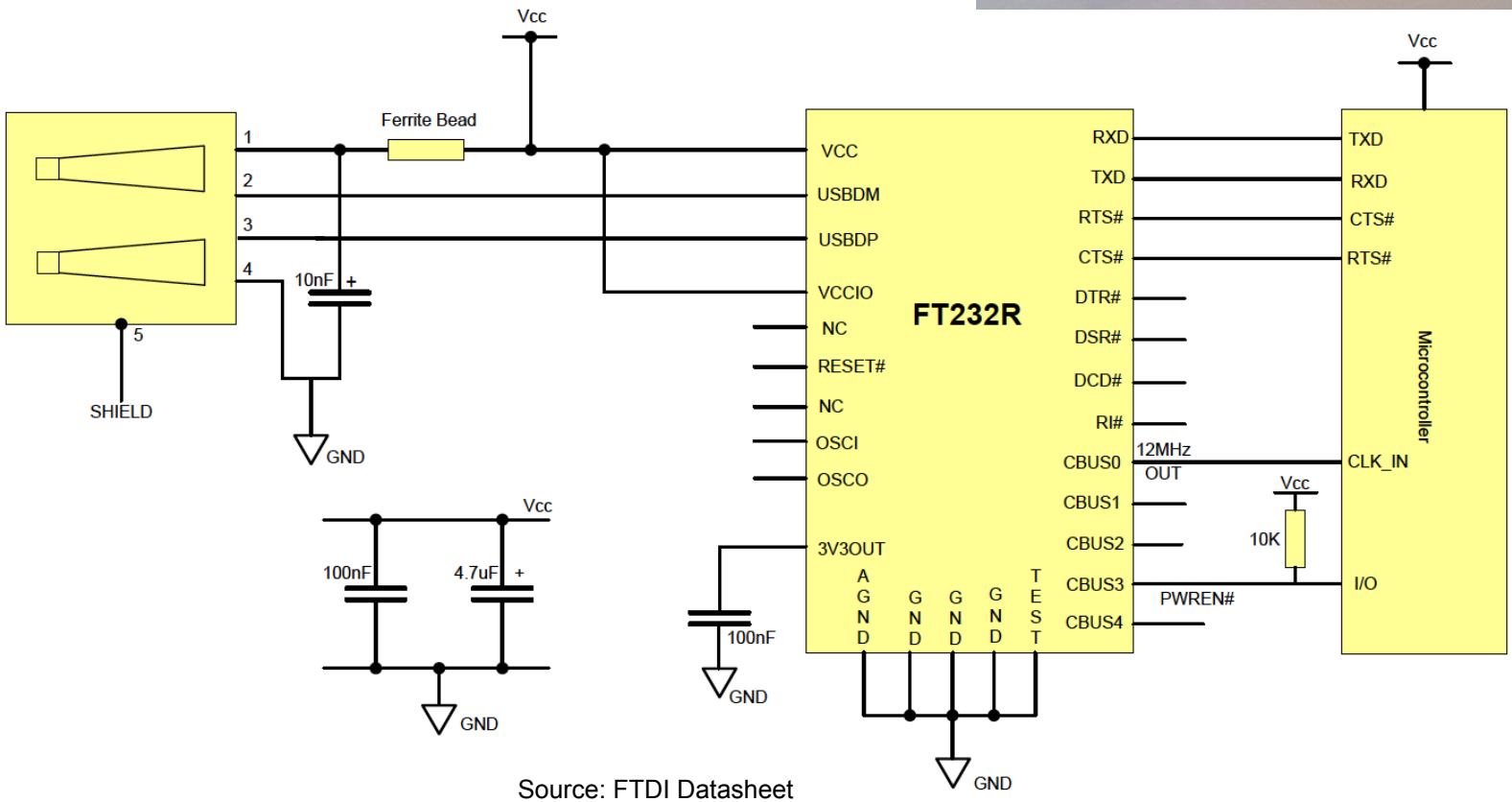


- Hardware handshake: Request-to-send, clear-to-send

USB to Serial UART
FTDI FT232RL

- UART = Universal Asynchronous Receiver/Transmitter
- http://www.ftdichip.com/Products/ICs/FT232R.htm





Source: FTDI Datasheet

FTDI FT232RL Virtual COM Port Drivers

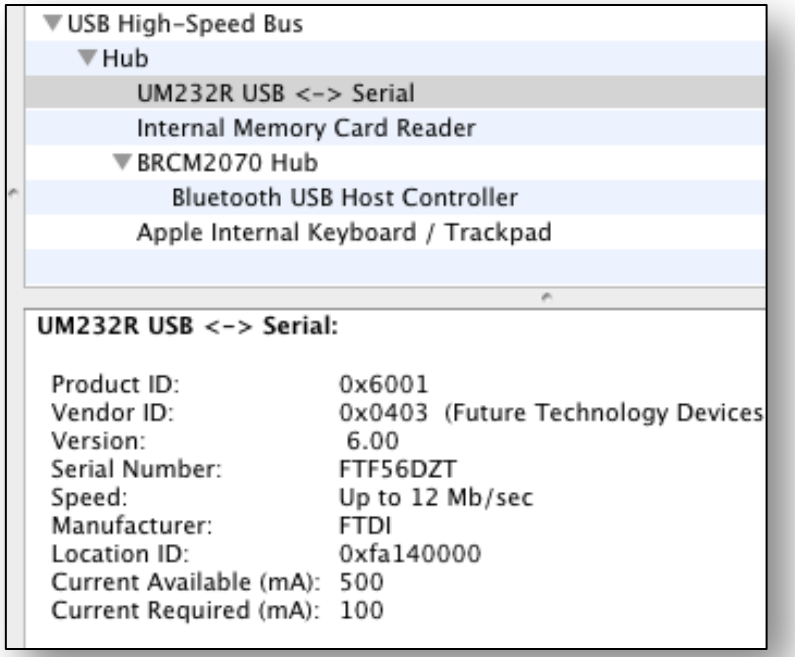
- Virtual COM port drivers
  - http://www.ftdichip.com/Drivers/VCP.htm
  - http://www.ftdichip.com/Support/Documents/AppNotes/AN_134_FTDI_Drivers_Installation_Guide_for_MAC_OSX.pdf
- USB device appears as virtual COM port

  cd /dev

  ls -l | grep usb

  cu.usbserial-FTF56DZT

  tty.usbserial-FTF56DZT
- Shows up in System Profiler



Use Virtual COM Port in Java

- Requires Java Serial Communications API
  - http://download.oracle.com/docs/cd/E17802_01/products/products/javacomm/reference/api/index.html
  - various implementations
- Windows
  - http://www.oracle.com/technetwork/java/index-jsp-141752.html
- Mac OS X, Linux
  - http://rxtx.qbang.org/wiki/index.php/Main_Page
  - http://users.frii.com/jarvi/rxtx/doc/
  - https://github.com/nyholku/purejavacomm
  - http://iharder.sourceforge.net/current/java/

Example: Sending to VCP in Java

```
import java.io.*;
import java.util.*;
import gnu.io.*;
public static void main(String[] args) throws Exception {
   Enumeration ports = CommPortIdentifier.getPortIdentifiers();
   while (ports.hasMoreElements()) {
      CommPortIdentifier portId = (CommPortIdentifier) ports.nextElement();
      if (portId.getPortType() == CommPortIdentifier.PORT_SERIAL) {
         if (portId.getName().equals("/dev/tty.usbserial-FTF56DZT")) {
            SerialPort sp = (SerialPort) portId.open("HelloWorld", 2000);
            OutputStream os = sp.getOutputStream();
            sp.setSerialPortParams(9600,
                                   SerialPort.DATABITS_8,
                                   SerialPort.STOPBITS_1,
                                   SerialPort.PARITY_NONE);
            os.write("hello world".getBytes());
            try { Thread.sleep(2000); } catch (InterruptedException ex) {}
            sp.close();
         }
} }
```

Direct Driver Interface

- Virtual COM Port (VCP) driver has limitations
- Full access to FT232R chip using proprietary driver
  - http://www.ftdichip.com/Drivers/D2XX.htm
- Example: Reading configuration data stored in EEPROM

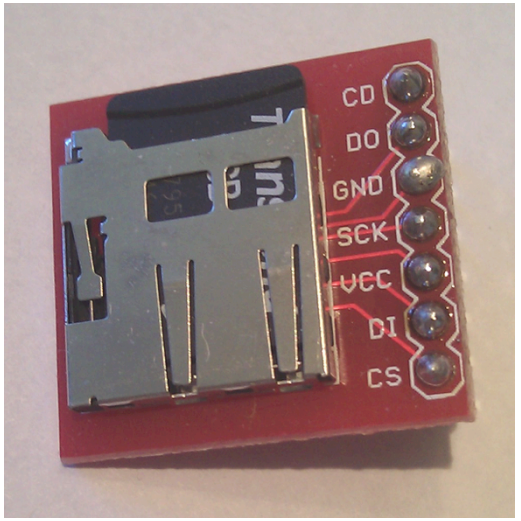
```
FT_STATUS ftStatus;
FT_HANDLE ftHandle0;
FT_PROGRAM_DATA Data;
FT_DEVICE ftDevice;
ftStatus = FT_Open(iport, &ftHandle0);
ftStatus = FT_GetDeviceInfo(ftHandle0, &ftDevice, NULL, ...);
ftStatus = FT_EE_Read(ftHandle0, &Data);
FT_Close(ftHandle0);
```

FT232R EEPROM Contents

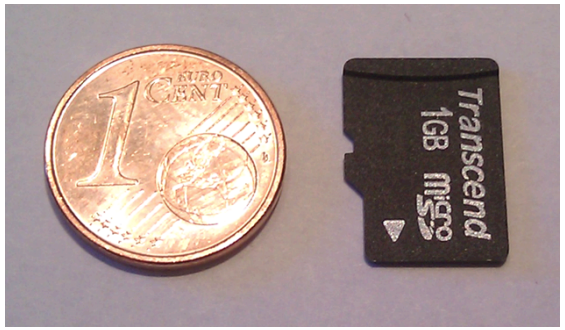
ftHandle0 = 0x100826400
Signature1 = 0
Signature2 = -1
Version = 2
VendorId = 0x0403
ProductId = 0x6001
Manufacturer = FTDI
ManufacturerId = FT
Description = UM232R USB <-> Serial
SerialNumber = FTF56DZT
MaxPower = 100
PnP = 1
SelfPowered = 0
RemoteWakeup = 1
UseExtOsc = 0x0
HighDriveIOs = 0x0
EndpointSize = 0x40

PullDownEnableR = 0x0
SerNumEnableR = 0x1
InvertTXD = 0x0
InvertRXD = 0x0
InvertRTS = 0x0
InvertCTS = 0x0
InvertDTR = 0x0
InvertDSR = 0x0
InvertDCD = 0x0
InvertRI = 0x0
Cbus0 = 0x2
Cbus1 = 0x3
Cbus2 = 0x1
Cbus3 = 0x1
Cbus4 = 0x5
RIsVCP = 0x0

Mass Data Storage



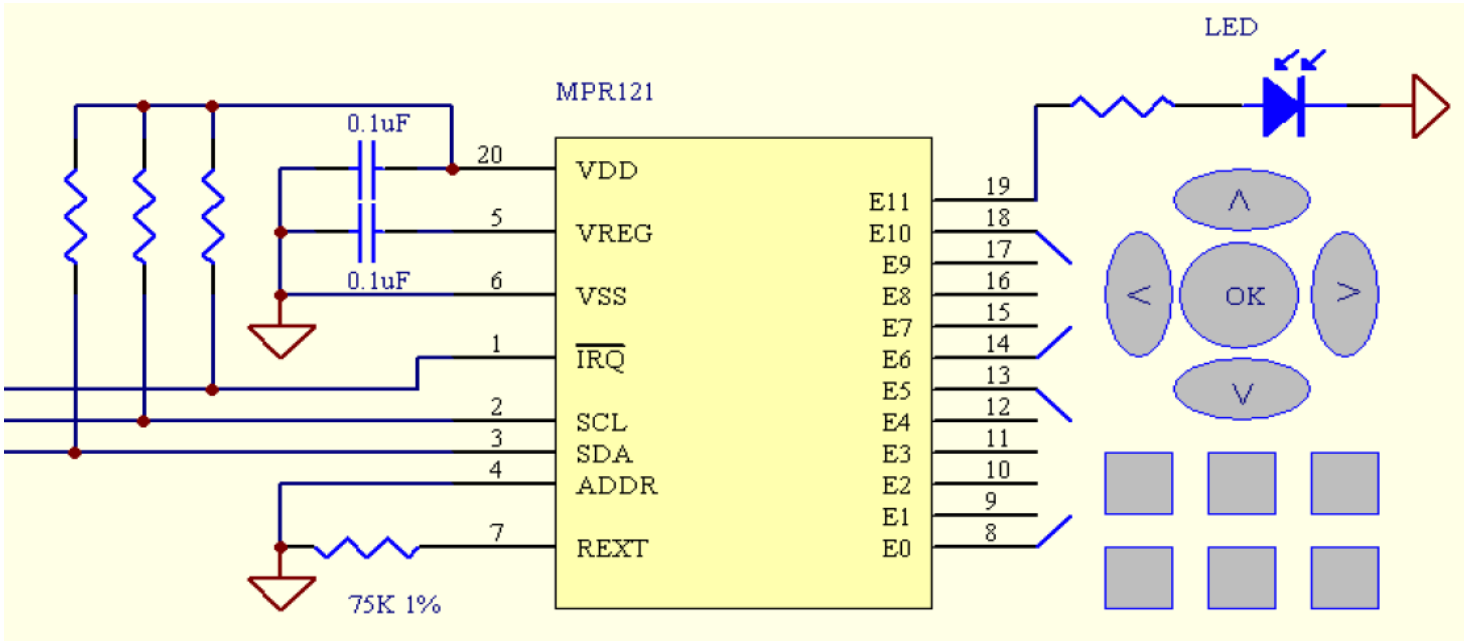
- Microcontrollers have extremely limited storage (EEPROMs for config. data)
- External I2C EEPROMs: still limited
- Memory cards: large capacity, easy to interface
- Example: Micro-SD cards
  - extended from MultiMediaCard (MMC)
  - max. 2GB for SDSC (Standard-Capacity)
  - file system: typically FAT16
  - communication: SPI (Serial Peripheral Interface)
  - V_{CC} = 3.3V, I = 20-100 mA
- Details: www.uni-koblenz.de/~physik/informatik/ECC/sd.pdf



TOUCH SENSORS

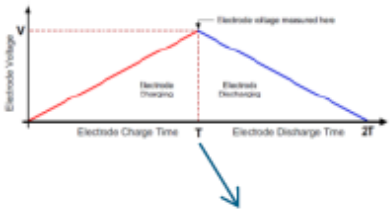
Capacitive Touch Sensor Controller Freescale Semiconductor MPR121

- VCC = 1.71..3.6 V
- I = 29 µA at 16 ms sampling interval
- 12 capacitance sensing inputs, connect to electrodes
- I2C interface



Source: Freescale Semiconductor Datasheet

Capacitive Touch Sensor Controller Freescale Semiconductor MPR121



CC/ Charge current & time control

Clock Gen Power control

To host

12 channels

Multiplex

10bit ADC

Filter

I2C logic

LED, GPIO control

auto configure, baseline tracking, touch/release detection

Cap Cb: ~10pf background Stray on input

C1 : ~100pf,Stray Cap to GND

Cap Cx: ~0.01pf induced by finger

C2 :10~100pf,Stray Cap to GND, or short to GND directly

Common Ground

Source: Freescale Semiconductor Datasheet

Capacitive Touch Sensor Controller Freescale Semiconductor MPR121

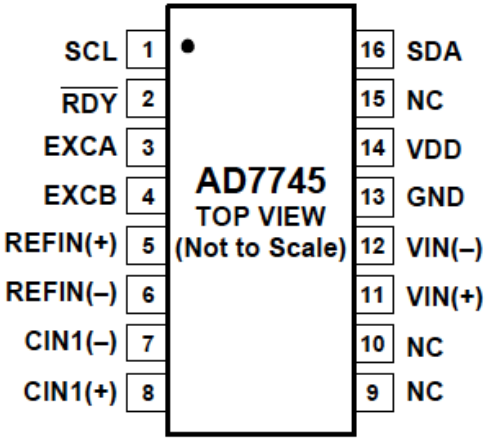
- Programmable charge current and charge time
  - I = 1..63µA, T = 0.5-63 µs

$$C = \frac{Q}{U} = \frac{I \cdot T}{U} \Leftrightarrow U = \frac{I \cdot T}{C}$$

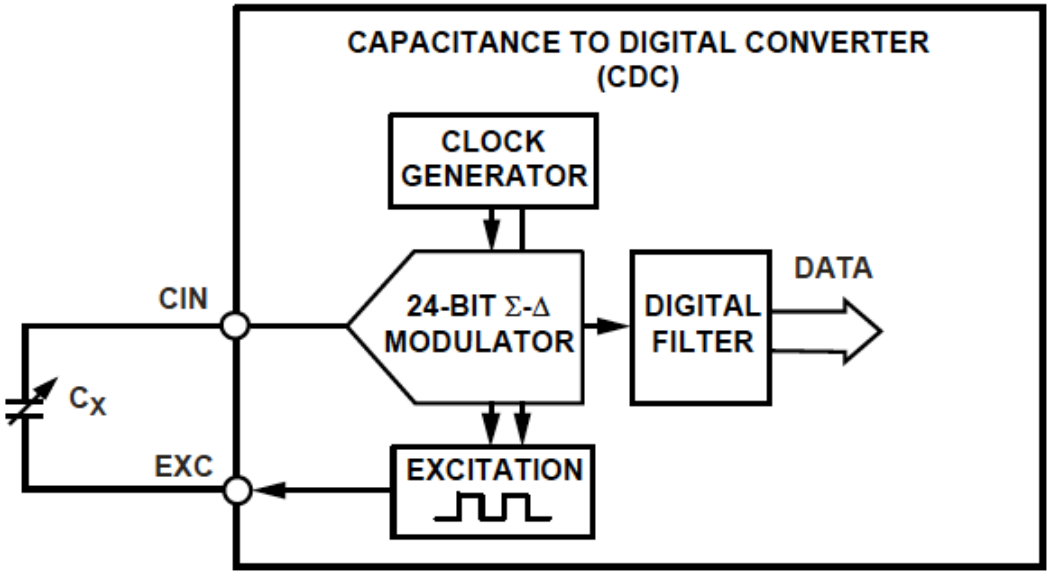
- ADC measures voltage after charge time
  - Measured voltage inversely proportional to capacitance
- Filters remove high and low frequency noise
- Can be configured for touch recognition
  - Debouncing, touch and release tresholds
- Auto-calibration
  - Continuously measures baseline capacitance

Capacitive Touch Sensor IC
Analog Devices AD7745/AD7746

- 24-Bit capacitance-to-digital converter
  - Temperature sensor
  - Update rate: 10 Hz to 90 Hz
  - V_{CC} = 2.7..5.25V, I = 0.7mA
- I2C interface
- Operation
  - EXC = square-wave excitation signal
  - CX = capacitance
  - CIN = modulator input
  - modulator samples charge, data filtered, scaled (calibration)



Source: FTDI Datasheet



Source: FTDI Datasheet

Registers

- read/write via I2C

Register	Address Pointer (Dec)	Address Pointer (Hex)	Dir	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				Default Value							
Status	0	0x00	R	-	-	-	-	EXCERR	RDY	RDYVT	RDYCAP
				0	0	0	0	0	1	1	1
Cap Data H	1	0x01	R	Capacitive channel data—high byte, 0x00							
Cap Data M	2	0x02	R	Capacitive channel data—middle byte, 0x00							
Cap Data L	3	0x03	R	Capacitive channel data—low byte, 0x00							
VT Data H	4	0x04	R	Voltage/temperature channel data—high byte, 0x00							
VT Data M	5	0x05	R	Voltage/temperature channel data—middle byte, 0x00							
VT Data L	6	0x06	R	Voltage/temperature channel data—low byte, 0x00							
Cap Setup	7	0x07	R/W	CAPEN	CIN2[1]	CAPDIFF	-	-	-	-	CAPCHOP
				0	0	0	0	0	0	0	0
VT Setup	8	0x08	R/W	VTEN	VTMD1	VTMD0	EXTREF	-	-	VTSHORT	VTCHOP
				0	0	0	0	0	0	0	0
EXC Setup	9	0x09	R/W	CLKCTRL	EXCON	EXCB	$\overline{\text{EXCB}}$	EXCA	$\overline{\text{EXCA}}$	EXCLVL1	EXCLVL0
				0	0	0	0	0	0	1	1
Configuration	10	0x0A	R/W	VTFS1	VTFS0	CAPFS2	CAPFS1	CAPFS0	MD2	MD1	MD0
				1	0	1	0	0	0	0	0
Cap DAC A	11	0x0B	R/W	DACAENA	DACA—7-Bit Value						
				0	0x00						
Cap DAC B	12	0x0C	R/W	DACBENB	DACB—7-Bit Value						
				0	0x00						
Cap Offset H	13	0x0D	R/W	Capacitive offset calibration—high byte, 0x80							
Cap Offset L	14	0x0E	R/W	Capacitive offset calibration—low byte, 0x00							
Cap Gain H	15	0x0F	R/W	Capacitive gain calibration—high byte, factory calibrated							
Cap Gain L	16	0x10	R/W	Capacitive gain calibration—low byte, factory calibrated							
Volt Gain H	17	0x11	R/W	Voltage gain calibration—high byte, factory calibrated							
Volt Gain L	18	0x12	R/W	Voltage gain calibration—low byte, factory calibrated							

Source: FTDI Datasheet

ATMEL QTouch

- Some AVRs integrate capacitive sensing hardware
  - Can also be implemented on standard microcontrollers
- ATMEL QTouch library
  - http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4627
  - http://www.atmel.com/dyn/resources/prod_documents/doc8207.pdf