ST7 MICROCONTROLLER TRAINING

1. INTRODUCTION
2. CORE
3. ADDRESSING MODES
4. ASSEMBLY TOOLCHAIN
5. STVD7 DEBUGGER
6. HARDWARE TOOLS
7. PERIPHERALS
8. ST-REALIZER II
9. C TOOLCHAINS
PURPOSE AND OBJECTIVES

• Purpose
  ➢ To improve your knowledge on standard ST7 peripherals

• Objectives
  ➢ List the main features of most of standard ST7 peripherals
  ➢ Optimize their use
  ➢ Develop an efficient code either in Assembly or C language using these peripherals
  ➢ To set up an application environment for a quick start
  ➢ Demonstrate peripherals performance using ST7 INDART demo board support
ST7 I/O PORTS

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 I/O PORTS
Overview

• All the I/Os are individually software configurable using 3 different registers:
  ➢ DDR: Data Direction Register
  ➢ DR: Data Register
  ➢ OR: Option Register

• ST72254: 22 multifunction bidirectional I/O lines
  ➢ 14 Standard I/Os (sink up to 5mA)
  ➢ 8 High Current I/Os (PA0-PA7 can sink up to 20mA)
  ➢ 6 Analog Inputs (PC0-PC5)
  ➢ 14 alternate Functions on 14 pins (for Timers, SPI and I2C)
  ➢ All the I/Os can be set-up as Interrupt inputs
## ST7 I/O PORT

**Safe I/O pin transition**

<table>
<thead>
<tr>
<th>DDR</th>
<th>OR</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Floating input</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Input pull-up with/without interrupt</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Output Open-Drain</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Output Push-Pull</td>
</tr>
</tbody>
</table>

### Reset State

- **00**
- **10**
- **11**
ST7 I/O PORT Basic structure

- Software selectable configuration
- HIGH FLEXIBILITY for software and PC board layout
**ST7 I/O PORT**

**Settings & electrical behaviour**

- Configuration given when no external Hardware is connected to the pins

<table>
<thead>
<tr>
<th>I/O Pin</th>
<th>Input Floating</th>
<th>Input Pull_up</th>
<th>Output Open Drain</th>
<th>Output Push-Pull</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>OR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Written DR</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>I/O Pin</td>
<td>Floating</td>
<td>Floating</td>
<td>Vdd</td>
<td>Vdd</td>
</tr>
<tr>
<td>Read DR</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Input Pull-up:**

- 0:關閉
- 1:開啟

**Output:**

- Push-Pull: 1
- Open Drain: 1
- Floating: 0

**Output Values:**

- Vdd: 1
- Vss: 0
- Floating: X

**Read DR Values:**

- X: 互斥
- 1: 确定
- 0: 不确定
ST7 EXTERNAL INTERRUPTS

- **Interrupt generation**
  - Negative edge only
  - Positive edge only
  - Positive and Negative edge
  - Negative edge and low level
  - Positive edge and high level

- **Edge/level selection**

- **Interrupt Source 1**
  - Negative edge only
  - Positive edge only
  - Positive and Negative edge
  - Negative edge and low level
  - Positive edge and high level

- **Interrupt Source 2**
  - Latched
  - Not Latched

- ST7 Interrupt Controller
ST7 I/O PORT Block Diagram

Note: Refer to the Port Configuration table for device specific information.
PROGRAMMING TIPS
I/O Port (1)

• AD conversion
  ➢ Each pin used by the ADC cell must be configured as floating input (i.e. without pull-up resistors) before activating the analog input mode (which is the usual default state)

• Alternate function
  ➢ A signal coming from an on-chip peripheral can be output on a port. In this case, the I/O is automatically configured in output mode.
  ➢ A signal coming from an I/O can be an input to an on-chip peripheral. In this case, it must be configured as Input without interrupt (Floating Input).
PROGRAMMING TIPS
I/O Port (2)

• Open Drain Outputs can be used for bus driving where several devices are connected on the same line (to avoid conflicts: lines can be pulled low or in high impedance). They can be wired together in parallel to increase current drive capability (I2C bus connections typically).

• Voltages driving an Analog Input should always stay within the absolute maximum ratings (Vss-0.3V to Vdd+0.3V)

• Pull-up resistors typically deliver 50µA under 5V

• The toggling time on any output pin will be approximately 30ns for a 50pF load
I/O Ports Configuration Example

- Fill the dedicated I/O port registers in order to have the following configuration:
  - PB0:PB2
    - Push-Pull Output (high level)
  - PB3, PB4
    - Floating Input
  - PB5
    - Input with Interrupt
  - PB6
    - Push-Pull Output (low level)
  - PB7
    - Output (High Impedance)
I/O Ports Configuration Example

- Fill the dedicated I/O port registers in order to have the following configuration:

- PB0:PB2
  - Push-Pull Output (high level)

- PB3, PB4
  - Floating Input

- PB5
  - Input with Interrupt

- PB6
  - Push-Pull Output (low level)

- PB7
  - Output (High Impedance)
ST7 EEPROM Data

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 EEPROM Overview

• Eeprom features:
  - Used to store non volatile data
  - On M5 devices, due to a multi-cycle write access, the EEPROM cannot use to execute code (time access: 1 cycle $\frac{1}{2}$ instead of 1)
  - On M6 devices (XFlash), code can be executed in EEPROM data.

• A parallel writing allows to program up to 32 bytes in a row

- Write cycle = 5 ms
- Write Erase cycles = 300 000 cycles
- Data retention = 20 years
ST7 EEPROM Block Diagram

**EEPCR**
- RESERVED
- E2ITE
- E2LAT
- E2PGM

**Interrupt Request**
- FALLING EDGE DETECTOR
  - HIGH VOLTAGE PUMP

**Address Decoder**
- ADDRESS BUS
  - 11
  - 5
  - 256
  - 5
  - 256

**Row Decoder**
- 1 ROW = 32 * 8 BITS

**Data Multiplexer**
- 32*8 BITS DATA LATCHES
  - DATA BUS
  - BUFFER
  - 8
  - 8

**EE2C**
- INTERRUPT REQUEST

**EE2IE**
- 0 int disabled
- 1 int enabled

**EE2LAT**
- 0 read mode
- 1 write mode

**EE2PGM**
- 0 prog finished or not started
- 1 start programming
EEPROM Programming Cycle

- **Write of data latches**
- **Erase cycle**
- **Write cycle**
- **Read operation not possible**
- **Read operation allowed**
- **INTRERRUPT REQUEST**
- **E2LAT**
- **E2PGM**
- **INTERNAL PROGRAMMING VOLTAGE**
- **Interrupt vector fetch**
PROGRAMMING TIPS
EEPROM

• A write is done by programming the e2prom control register:
  ➢ Step 1: set E2LAT bit in order to select the write mode
  ➢ Step 2: write up to 32 bytes with 11 MSbits in common
  ➢ Step 3: set E2PGM bit to start the programming cycle
  ➢ Step 4: wait for E2PGM reset ie end of the programming cycle

• A read is performed as if it was a read only memory: no software overhead required!

• The halt instruction or a reset immediatly stop any eeprom operation. The wait mode has no effect
ST7 EEPROM Programming mode

• To read any memory location, E2LAT must be cleared
• To program EEPROM:

  BSET EEPCR, #E2LAT ; Set E2LAT bit
  LD VARIABLE_i, A ; Write to up to 32 locations with
  ... ; the same 11 address MSB

  ; BSET EEPCR, #E2ITE ; Enable interrupts if needed
  BSET EEPCR, #E2PGM ; Start programming
  .wait
  BTJT EEPCR, #E2PGM, wait ; Wait for end of write cycle
ST7 A/D CONVERTER

• Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 A/D CONVERTER

Overview (1)

• 8-Bit and 10-Bit ADC depending on the device:
  ➢ 8-Bit on ST72254, ST72334, ST7Lite0
  ➢ 10-Bit on ST7Lite2, ST7Lite3, ST7Lite1B, ST7UltraliteS, ST72264, ST72344, ST72521, ST7262

• Conversion based on successive approximations with sample and hold circuitry

• Up to 16 analog channels

• Conversion time at max ADC frequency ($f_{\text{ADC}}=4\text{MHz}$):
  ➢ 8-bit $\rightarrow$ 3$\mu$s
  ➢ 10-Bit $\rightarrow$ 3.5$\mu$s or 7.5$\mu$s (but for $F_{\text{ADC max}} = 2\text{MHz}$) depending on the devices

• Integrated op-amp for a zoom function (Lite family):
  ➢ Gain: 8
  ➢ Range: [0-250mV] or [0-430mv] depending on devices
ST7 A/D CONVERTER
Overview (2)

- ADON bit (ADC on/off bit) allows to reduce power consumption.
- EOC or COCO bit indicates to the MCU that the conversion is completed
- Special features:
  - x8 Amplifier for zooming (ST7Lite family)
  - End of conversion interrupt generation (ST7Hub)
  - Average
  - Speed adjustment
  - One shot mode (ST7262)
ST7 AD CONVERTER
Overview (3)

• ADC vs. low consumption modes
  ➢ Wait mode doesn't affect the ADC
  ➢ Halt mode stops the ADC

• Conversion results vs. analog input voltage
  ➢ Conversion is linear
  ➢ If analog voltage input > \( V_{REF} \):
    converted result = FFh (no overflow indication)
  ➢ If analog voltage input < \( V_{SS} \):
    converted result = 00h (no underflow indication)

• Two voltage reference configuration:
  ➢ \( V_{REF} \) and \( V_{SSA} \) available:
    ✓ \( 0.7 \times V_{DD} < V_{REF} < 5.5V \) with decoupling capacitors.
  ➢ \( V_{REF} \) and \( V_{SSA} \) pins not available:
    ✓ connection done internally to \( V_{DD} \) and \( V_{SS} \)
ST7 A/D CONVERTER
Block diagram

- MUX
- SAMPLE & HOLD
- ANALOG TO DIGITAL CONVERTER

Control Status Register (CSR)
- EOC
- Speed
- ADON
- Slow
- CH0
- CH1
- CH2
- CH3

Data Register High (DR)
- AD9
- AD8
- AD7
- AD6
- AD5
- AD4
- AD3
- AD2

Data Register Low (DR)
- AD1
- AD0

Channels:
- AIN0
- AIN1
- AIN15
- AIN16

Conversion Rates:
- Fadc
- Fcpu
- x1
- 1/2
- 1/4
ST7 A/D CONVERTER
PROGRAMMING TIPS

• Procedure:
  - Step 1: Configure the pin to be used as the analog input in input floating mode (no pull-up, no interrupt)
  - Step 2: Select the channel to be converted using the bits CH[3:0] in CSR register and set the ADON bit
  - Step 3: Wait until COCO or EOC bit is set. A continuous conversion is performed.
  - Step 4: Read the ADC data registers.

• To reach the best accuracy, the impedance seen by the analog input pin must be lower than 10Kohm.
## ST7 A/D CONVERTER
### ADC TYPE VS. DEVICE

<table>
<thead>
<tr>
<th>ADC</th>
<th>MCU</th>
<th>ST7Lite0</th>
<th>ST7Lite2</th>
<th>ST7F264</th>
<th>ST7F324</th>
<th>ST7F521</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC TYPE</td>
<td>8-Bit</td>
<td>10-Bit</td>
<td>10-Bit</td>
<td>10-Bit</td>
<td>10-Bit</td>
<td></td>
</tr>
<tr>
<td>ERROR (LSB)</td>
<td>±2</td>
<td>±4</td>
<td>±4</td>
<td>±4</td>
<td>±4</td>
<td></td>
</tr>
<tr>
<td>NB. INPUTS</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>8</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>
• Which are the two types of ST7 ADC?
  ➢ 8-Bit & 10-Bit

• What is the configuration of the analog input pin?
  ➢ The pin used as an analog input must be configured in input floating mode (no pull-up, no interrupt).

• Which bit indicates the completion of the conversion?
  ➢ The EOC or COCO bit. It is set by hardware and cleared by software.

• What is the conversion time of the 10-Bit ADC?
  ➢ The conversion time is 7.5µs at $f_{ADC}=2$MHz for some devices and 3.5µs for some others at $f_{ADC}=4$MHz.
Fill ADCCSR register in order to have an analog conversion on AIN4 with $f_{CPU}/2$ (Refer to ST7Lite0 datasheet).
ADC Configuration Example

- Fill ADCCSR register in order to have an analog conversion on AIN4 with $f_{CPU}/2$ (Refer to ST7Lite0 datasheet).

<table>
<thead>
<tr>
<th>COCO</th>
<th>ADON</th>
<th>CH3</th>
<th>CH2</th>
<th>CH1</th>
<th>CH0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

To be tested
Exercise N°4
ADC

• Purpose
  ➢ Be familiar with the ADC peripheral manipulation

• Objectives
  ➢ Step 1: Convert the analog input PB3 and display the conversion result on PA[0:4] and PB[0:2] with PA LSB and PB MSB.

  ➢ Step 2: Convert the analog input PB3 and display it as follows:
    ✓ 0V < V_{in} < 1.5V → LED on PB0 is ON
    ✓ 1.5V < V_{in} < 3.5V → LED on PB1 is ON
    ✓ 3.5V < V_{in} < 5V → LED on PB2 is ON

Refer to: Training Manual: Exercise 4 – P.26/30
ST7 ANALOG COMPARATOR

• Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 Analog comparator

• Features (lite1b device):
  ➢ External and internal reference voltage (1.2V fixed internal bandgap or 16 voltage levels between 3.2V and 0.2V with a 0.2V step)
  ➢ Output available externally or possibility to connect it internally to the break function of the 12-bit ARTimer
  ➢ COMPIN+ is also connected to AIN0 and COMPIN- to AIN4
  ➢ If COMPIN+ < COMPIN- then COMPOUT=0
  ➢ If COMPIN+ > COMPIN- then COMPOUT=1
  ➢ Stabilization time of 500ns
  ➢ CINV bit to invert comparator output (COMPOUT)
  ➢ Dedicated interrupt on output change
ST7 SCI

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 SCI - New features

- Division by 2 has been removed from Rx and Tx formula on new devices
- New features: parity control and reduced power consumption mode

<table>
<thead>
<tr>
<th>New devices (X and HD flash)</th>
<th>Other devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST7226x</td>
<td>ST72511</td>
</tr>
<tr>
<td>ST7232x</td>
<td>ST72511</td>
</tr>
<tr>
<td>ST72521</td>
<td>ST72532</td>
</tr>
<tr>
<td>ST7262</td>
<td>ST72334</td>
</tr>
<tr>
<td>ST7263B</td>
<td>ST72314</td>
</tr>
<tr>
<td>ST7234x</td>
<td>ST72124</td>
</tr>
<tr>
<td></td>
<td>ST72589</td>
</tr>
<tr>
<td></td>
<td>ST72389</td>
</tr>
<tr>
<td></td>
<td>ST7263</td>
</tr>
<tr>
<td></td>
<td>ST7285</td>
</tr>
<tr>
<td></td>
<td>ST72171</td>
</tr>
<tr>
<td></td>
<td>ST72121</td>
</tr>
<tr>
<td></td>
<td>ST72331</td>
</tr>
</tbody>
</table>
ST7 SCI Overview (1)

• Full duplex, asynchronous communication, 2 pins (RDI / TDO)

• Dual baud rate generator (maximum speed for SCI TX and RX: 500kHz or 500000 bauds)

\[
F_{tx} = \frac{F_{cpu}}{[16 \times PR] \times TR} \quad F_{rx} = \frac{F_{cpu}}{[16 \times PR] \times RR}
\]

• Programmable word length
  ✓ 8 bits
  ✓ 9 bits

• Noise, overrun and frame error detection
• Parity control (new devices only!)
• Reduced power consumption mode (new devices only!)
ST7 SCI Overview (2)

• Muting functions for multiprocessor configurations

• Receiver wake function by the most significant bit or idle line

• 6 Interrupt sources with flags
  ➢ Transmit data register empty
  ➢ Transmission complete
  ➢ Received data ready to be read
  ➢ Overrun error detected
  ➢ Parity error (*new devices only!*)

• 2 Flags without interrupt
  ➢ Noise flag
  ➢ Framing error
ST7 SCI - Serial data format

- **8-bit word length**

  - Data frame
  - Possible parity bit
  - Start
  - Bit 0 Bit 1 Bit 2 Bit 3 Bit 4 Bit 5 Bit 6 Bit 7 Stop
  - Idle frame
  - Break frame
  - Extra ‘1’

- **9-bit word length**

  - Data frame
  - Possible parity bit
  - Start
  - Bit 0 Bit 1 Bit 2 Bit 3 Bit 4 Bit 5 Bit 6 Bit 7 Bit 8 Stop
  - Idle frame
  - Break frame
  - Extra ‘1’
SCI Sampling Data Format

- Each bit time is
  - Divided by 16 by the SCI clock
  - Sampled 3 times on the 8th, 9th and 10th count of the SCI clock

- NF flag is set if the 3 sampling are not equal but the reception is still available

<table>
<thead>
<tr>
<th>Data Sampled values</th>
<th>Received bit value</th>
<th>NF Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
SCI Block Diagram

Transmit data register
Transmit shift register
Receive data register
Receive shift register
Transmit Control
Wake-Up Unit
Receive Control
Control register 1
Control Register 2
Status Register
SCI Interrupt Control
Transmit rate Control
Receive rate Control
f_{cpu} / 2 / 16 / PR

Removed on new devices

RDI pin
TDO pin
ST7 SCI Clock selection

- **EXTENDED PRESCALER TRANSMITTER RATE CONTROL**
- **EXTENDED TRANSMITTER PRESCALER REGISTER**
- **EXTENDED RECEIVER PRESCALER REGISTER**
- **EXTENDED PRESCALER RECEIVER RATE CONTROL**
- **CONVENTIONAL BAUD RATE GENERATOR**

- **f_CPU**
  - /16
  - /2
  - /PR

- **ETPR**
- **ERPR**

- **TRANSMITTER RATE CONTROL**
- **RECEIVER RATE CONTROL**

- **BRR**

- **Removed on new devices**
SCI Configurable Baud Rate (1/2)

- **New devices**
  - Conventional Baud Rate Generation
    \[ F_{tx} = \frac{F_{cpu}}{[16 \times PR] \times TR} \]
    \[ F_{rx} = \frac{F_{cpu}}{[16 \times PR] \times RR} \]
  - Extended Baud Rate Generation
    \[ F_{tx} = \frac{F_{cpu}}{[16 \times PR] \times TR \times ETPR} \]
    \[ F_{rx} = \frac{F_{cpu}}{[16 \times PR] \times RR \times ERPR} \]

- **Others devices**
  - Conventional Baud Rate Generation
    \[ F_{tx} = \frac{F_{cpu}}{[16 \times PR \times 2] \times TR} \]
    \[ F_{rx} = \frac{F_{cpu}}{[16 \times PR \times 2] \times RR} \]
  - Extended Baud Rate Generation
    \[ F_{tx} = \frac{F_{cpu}}{16 \times ETPR} \]
    \[ F_{rx} = \frac{F_{cpu}}{16 \times ERPR} \]
### SCI Configurable Baud Rate (2/2)

**New devices**

<table>
<thead>
<tr>
<th>TR - SCT2:SCT0</th>
<th>PR - SCP1,SCP0</th>
<th>ETPR</th>
<th>Baud rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 - 110</td>
<td>13 - 11</td>
<td>0</td>
<td>600</td>
</tr>
<tr>
<td>16 - 100</td>
<td>13 - 11</td>
<td>0</td>
<td>2400</td>
</tr>
<tr>
<td>8 - 011</td>
<td>13 - 11</td>
<td>0</td>
<td>4800</td>
</tr>
<tr>
<td>4 - 010</td>
<td>13 - 11</td>
<td>0</td>
<td>9600</td>
</tr>
<tr>
<td>2 - 001</td>
<td>13 - 11</td>
<td>0</td>
<td>19200</td>
</tr>
<tr>
<td>1 - 000</td>
<td>13 - 11</td>
<td>0</td>
<td>38400</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>13</td>
<td>38400</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>9</td>
<td>57600</td>
</tr>
</tbody>
</table>

Values given for fCPU =8MHz

**Other devices**

<table>
<thead>
<tr>
<th>TR - SCT2:SCT0</th>
<th>PR - SCP1,SCP0</th>
<th>ETPR</th>
<th>Baud rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 - 110</td>
<td>13 - 11</td>
<td>0</td>
<td>300</td>
</tr>
<tr>
<td>16 - 100</td>
<td>13 - 11</td>
<td>0</td>
<td>1200</td>
</tr>
<tr>
<td>8 - 011</td>
<td>13 - 11</td>
<td>0</td>
<td>2400</td>
</tr>
<tr>
<td>4 - 010</td>
<td>13 - 11</td>
<td>0</td>
<td>4800</td>
</tr>
<tr>
<td>2 - 001</td>
<td>13 - 11</td>
<td>0</td>
<td>9600</td>
</tr>
<tr>
<td>1 - 000</td>
<td>13 - 11</td>
<td>0</td>
<td>19200</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>13</td>
<td>38400</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>9</td>
<td>57600</td>
</tr>
</tbody>
</table>

Values given for fCPU =8MHz
SCI Configuration Example

- Fill the SCI registers in order to configure the SCI cell in:
  - 8 Bit word reception at 9600 Bauds
  - 8 bit word transmission at 1200 Bauds
  - Interrupt generation when RDRF is set (reception flag)
- \( f_{CPU} = 8\text{MHz} \)
SCI Configuration Example

- Fill the SCI registers in order to configure the SCI cell in
  - 8 Bit word reception at 9600 Bauds
  - 8 bit word transmission at 1200 Bauds
  - Interrupt generation when RDRF is set (reception flag)

- fCPU = 8MHz
Serial Peripheral Interface

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 SPI
Overview (1/2)

• The SPI cell allows a full duplex synchronous serial communication between 2 devices

• Main features:
  ➢ Full duplex, 3 wire synchronous transfers
  ➢ Master: 6 frequency available. It rates up to 2 MHz
  ➢ Slave mode: it rates up 4 MHz
  ➢ /SS (Slave Select) also configurable through software (MISCR2 or SPICSR on Xflash and HDFlash devices)

• The clock is programmable: polarity and phase
3 Interrupt sources with flags
- Data transfer: data transfer completed
- Overrun error: SPIF bit has not been cleared *(new devices only!)*
- Fault flag: fault in master mode detected

1 Flag without interrupt
- Write collision: access to SPIDR during a transmission
ST7 SPI
Master-Slave communication

Master
- 8-bit Shift Register
- SPI Clock Generator
- MISO
- MOSI
- SCK
- SS
- 5V

Slave
- 8-bit Shift Register
- MISO
- MOSI
- SCK
- SS
- 5V
ST7 SPI
Block diagram
ST7 SPI
Single master configuration
**SPI Configuration Example**

- Fill the SPICR register in order to configure the SPI cell in:
  - **Master mode**
  - **Serial clock at 500KHz** (fCPU=8MHz)
  - **Sampling on 2nd edge**
  - **High level after clock signal**
  - **No interrupt generation**

<table>
<thead>
<tr>
<th>SPIE</th>
<th>SPE</th>
<th>SPR2</th>
<th>MSTR</th>
<th>CPOL</th>
<th>CPHA</th>
<th>SPR1</th>
<th>SPR0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SPI Configuration Example

- Fill the SPICR register in order to configure the SPI cell in:
  - **Master mode**
  - **Serial clock at 500KHz** (fCPU=8MHz)
  - **Sampling on 2nd edge**
  - **High level after clock signal**
  - **No interrupt generation**
ST7 I2C

• Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
ST7 I2C Overview

• The I2C cell provides all I2C bus specific sequencing, protocol, arbitration and timing in order to reduce as much as possible the software overhead

• Polling Management or Interrupt Driven Cell

• Main features:
  ➢ Multi Master capability
  ➢ Interrupt generation
  ➢ Standard I2C mode (up to 100kHz) and Fast I2C mode (up to 400kHz)
  ➢ 7-bit and 10-bit addressing
I2C Bus Protocol

Diagram showing the I2C protocol with SDA and SCL lines, MSB and ACK conditions, and start and stop conditions.
I2C Master Mode
Transfer Sequencing

7-bit Master Transmitter

<table>
<thead>
<tr>
<th>S</th>
<th>Address</th>
<th>A</th>
<th>DATA1</th>
<th>A</th>
<th>DATA2</th>
<th>A</th>
<th>DATA n</th>
<th>A</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>EV5</td>
<td></td>
<td>EV6</td>
<td>EV8</td>
<td>EV8</td>
<td>EV8</td>
<td></td>
<td>EV8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7-bit Master Receiver

<table>
<thead>
<tr>
<th>S</th>
<th>Address</th>
<th>A</th>
<th>DATA1</th>
<th>A</th>
<th>DATA2</th>
<th>A</th>
<th>DATA n</th>
<th>NA</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>EV5</td>
<td></td>
<td>EV6</td>
<td></td>
<td>EV7</td>
<td>EV7</td>
<td></td>
<td>EV7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

EV5: EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.
EV6: EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).
EV7: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.
EV8: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

Send by the master
Send by the slave
I2C Slave Mode
Transfer Sequencing

7-bit Slave Receiver

<table>
<thead>
<tr>
<th>S</th>
<th>Address</th>
<th>A</th>
<th>DATA1</th>
<th>A</th>
<th>DATA2</th>
<th>A</th>
<th>DATA n</th>
<th>A</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EV1</td>
<td></td>
<td>EV2</td>
<td></td>
<td>EV2</td>
<td></td>
<td>EV2 EV4</td>
</tr>
</tbody>
</table>

7-bit Slave Transmitter

<table>
<thead>
<tr>
<th>S</th>
<th>Address</th>
<th>A</th>
<th>DATA1</th>
<th>A</th>
<th>DATA2</th>
<th>A</th>
<th>DATA n</th>
<th>NA</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EV1</td>
<td></td>
<td>EV3</td>
<td></td>
<td>EV3</td>
<td></td>
<td>EV3 EV4</td>
</tr>
</tbody>
</table>

EV1: EVF=1, ADSL=1, cleared by reading SR1 register.
EV2: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.
EV3: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.
EV4: EVF=1, STOPF=1, cleared by reading SR2 register.
I2C Clock Control

I2C CLOCK CONTROL REGISTER (CCR)

<table>
<thead>
<tr>
<th>FM/SM</th>
<th>CC6</th>
<th>CC5</th>
<th>CC4</th>
<th>CC3</th>
<th>CC2</th>
<th>CC1</th>
<th>CC0</th>
</tr>
</thead>
</table>

FM/SM : Fast / standard I2C mode
0: Standard I2C mode
1: Fast I2C mode
CC6-CC0 : 7bit clock divider

Standard Mode

\[ Fscl = \frac{F_{cpu}}{(2 \times [CC6..CC0]+2)} \]

Fast Mode

\[ Fscl = \frac{F_{cpu}}{(3 \times [CC6..CC0]+2)} \]
ST7 16-bit TIMER

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
16-bit TIMER
Overview (1)

• 16-bit free running counter driven by a software configurable prescaler
• 4 different modes:
  ➢ Input capture (2 pins): to latch the value of the counter after a transition on the ICAPi pin
  ➢ Output compare (2 pins): to control an output waveform or to indicate when a period of time is over
  ➢ One pulse: generation of a pulse when an external event occurs
  ➢ PWM: generation of a signal with frequency and pulse length set by software (OCR1 and OCR2)
The timer clock can be provided by:

- The internal clock with a configurable ratio
- An external source: F_{ext} must 4 times slower than the internal clock (ie \( F_{\text{max}} = 2\, \text{MHz} \))

<table>
<thead>
<tr>
<th>CC1</th>
<th>CC0</th>
<th>Timer clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>F_{cpu}/4</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>F_{cpu}/2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>F_{cpu}/8</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>External</td>
</tr>
</tbody>
</table>
16-bit TIMER
Input capture (1)

- Captures the counter value upon input signal edge detection
- Allows an external pulse length measurement
- Internal safety process in case of critical interrupts timing
16-bit TIMER
Input capture (1)
16-bit TIMER
Output compare (1)

- Event generation (Interrupt request/bit toggling) whenever the compare register matches the counter
- Indicates a period of time has elapsed and controls an output waveform
- Internal safety process in case of critical interrupts timing
16-bit TIMER
Output compare (2)
16-bit TIMER
Real Time Clock

• In each Interrupt Routine the OCR Register content is updated.
• There is no shift time (the counter is never reset externally).
16-bit TIMER
One pulse mode (1)

• Generation of a pulse synchronized with an external event

• Allows Phase Locked Loop Generation

• On Input Capture event
  ➢ The counter is reset
  ➢ The timer output pin is toggled

• On Output compare event
  ➢ The timer output pin is toggled
  ➢ The timer waits for the next Input Capture event
16-bit TIMER
One pulse mode (2)

- When a external event occurs on ICAP1 pin
- When the free running counter reaches OC1R register value
  - Free running counter is initialized to FFFCh
  - OLVL2 bit level is applied on the OCMP1 pin
  - ICF1 bit is set
  - OLVL1 bit level is applied on the OCMP1 pin
16-bit TIMER
One pulse mode (3)
16-bit TIMER
PWM mode (1)

• Automatic generation of a Pulse Width Modulated signal

• Period & pulse length set by software:
  ➢ The first Output Compare Register OC1R contains the length of the pulse
  ➢ The second Output Compare Register OCR2 contains the period of the pulse

• Resolution up to 100 steps at 20 KHz (fCPU =4 MHz): 1% of accuracy on the duty cycle
ST7 TIMER
PWM mode (2)

When the free running counter reaches OC2R register value

Free running counter is initialized to FFFCh
OLVL2 bit level is applied on the OCMP1 pin
ICF1 bit is set

When the free running counter reaches OC1R register value

OLVL1 bit level is applied on the OCMP1 pin
ST7 TIMER
PWM mode (3)

FREE RUNNING COUNTER VALUE

T_{\text{max}} = T_{\text{timer}} \times 65535

0000h
FFFFh
FFFCh
Compare 2
Compare 1

OLVL1=0
OLVL2=1

OCMP1
Output Compare pin
Timer output
PROGRAMMING TIPS
16-bit timer (1)

• Define Input capture pins as inputs through the corresponding Data Direction Register

• Read MSB first and then the LSB
  ➢ The counter LSB is buffered during the MSB read
  ➢ The counter LSB read accesses the buffered value
  ➢ Any access to the high byte disables the corresponding timer function until the low byte is accessed
  ➢ Disable the interrupts during any word access

• Writing the counter LSB resets the timer at FFFCh
  ➢ Any write access is valid (CLR TACLR or LD TACLR,A)
PROGRAMMING TIPS
16-bit timer (2)

• Clearing a status bit is performed by a read access to the status register followed by an access (read or write) to the low byte of the corresponding register

• The alternate counter register is always matching the counter

• Use the alternate counter register when you do not want to clear the Timer Overflow Flag

• No interrupt is generated on compare when the PWM is active, but the ICF1 bit is set every period and can generates an interrupt

• Be aware that the implicit reading performed by the emulator might clear the status flags
PROGRAMMING TIPS
16-bit timer (3)

Read

CHR → CLR buffered

ACHR → ACLR buffered

Any others Instructions

CLR → Returns the CLR buffered value
    Clear TOF bit

AChr → Returns the ACLR buffered value

Write

CLR → Reset counter to FFFCh
    Clear TOF bit

AChr → Reset counter to FFFCh
Timer Configuration Example

- Fill the Timer registers in order to generate a real time clock at 5ms using an interrupt strategy & a timer clock at 1µs (fCPU = 8MHz).
  - An interrupt is generated every 5ms using Output compare1.
  - OCMP1 Pin has to be toggled every period

- What is the value to add to the TAOC1HR & TAOC1LR every period?
Timer Configuration Example

- Fill the Timer registers in order to generate a real time clock at 5ms using an interrupt strategy & a timer clock at 1µs (fCPU = 8MHz).
  - An interrupt is generated every 5ms using Output compare1.
  - OCMP1 pin has to be toggled every period
- What is the value to add to the TAOCIHR & TAOC1LR every period?

\[ \text{5ms/1µs} = 5000 \Rightarrow 0x1388 \]
Lite Timers

• Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
Lite Timers Overview

• LT (Lite0,Ultralite):
  ➢ 1 free running 8-bit (Lite0) or 13-bit upcounter (Ultralite): 1
    1ms and 2ms timebase period (@Fosc=8MHz, overflow when
    the counter reaches $F9)
  ➢ Maskable timebase interrupt
  ➢ Input capture with dedicated interrupt (wake up from Halt
    capability)
  ➢ Watchdog: 2ms timeout @Fosc=8MHz, 1 bit to set (WDGD)
    to avoid a reset. WDG reset can be forced by setting WDGRF
    bit. WDG is enabled through WDGE (software WDG) or
    through option byte (hardware WDG).

• LT2 (Lite1b,Lite2,Lite3):
  ➢ 2 free running 8-bit upcounters: 1 with 1ms and 2ms timebase
    and the other with programmable timebase period from 4µs to
    1.024ms in steps of 4µs (@Fosc=8MHz)
  ➢ 2 maskable timebase interrupts
  ➢ Input capture with dedicated interrupt (wake up from Halt
    capability)
8-bit Auto Reload Timer

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
Overview
8-bit ARTimer

- Programmable Timer frequency (Fcounter)
- External Clock / counter event capability
- Up to 4 independant PWM signals (same frequency)
- Output compare and timebase Interrupts
- Up to 2 Input capture Channels with Interrupts
- Timer implemented on ST72171, ST72511R, ST72311R, ST72F521, ST7FLITE0, ST7FLITEUS
8-bit ARTimer
Fcounter definition

- Finput could be Fcpu or Fext.
- Fext Max = Fcpu/2.
- The Counter is incremented on rising edge of Finput.
- Fcounter is a division of Finput, according the prescaler driven by the CC2-CC0 bits of the Control Register.

<table>
<thead>
<tr>
<th>Fcounter</th>
<th>Finput=8MHz</th>
<th>CC2</th>
<th>CC1</th>
<th>CC0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finput</td>
<td>8MHz</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Finput /2</td>
<td>4MHz</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Finput /4</td>
<td>2MHz</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Finput /8</td>
<td>1MHz</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Finput /16</td>
<td>500KHz</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Finput /32</td>
<td>250KHz</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Finput /64</td>
<td>125KHZ</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Finput /128</td>
<td>62.5KHZ</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
8-bit ARTimer
Autoreload & PWM Outputs

### Counter value
- **DCRx Duty Cycle Register**: FF
- **ARR Auto Reload Register**: 00

**PWMx Output Pin**:
- OEx bit=1
- OPx bit=1

#### DCRx Value
<table>
<thead>
<tr>
<th>DCRx Value</th>
<th>PWMx Pin (OPx=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARR-1</td>
<td>Low Level</td>
</tr>
<tr>
<td>ARR</td>
<td>1 pulse = Tcounter</td>
</tr>
<tr>
<td>FFh</td>
<td>High Level</td>
</tr>
</tbody>
</table>
8-bit ARTimer
Input Capture

CFx bit is set
Interrupt generation
if CIEx bit is set

- Fcounter
- Counter Register
- ARTICx Pin
- Input Capture x Pin
- ICRx
- Input Capture x Register

01h 02h 03h 04h 05h 06h 07h
CSx bit=1

xxh 04h
8-bit ARTimer
Programming Tips

• Clearing the OVF (Overflow) bit is performed by reading the CSR register.

• Clearing the CFx (Capture Flag X) bit is performed by reading the ICRx register.

• In HALT mode, the Input capture Interrupt can be used to wake up the ST7.

• In HALT mode, the Overflow Interrupt can be used to wake up the ST7, if the external clock (Fext) is used. (external event detector mode).

• Be aware that the implicit reading performed by the emulator might clear the status flags.
12-bit Auto Reload Timers

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
Overview (1/2)
12-bit ARTimers

• AT (lite0, ultralite):
  ➢ No external pins, clock is Fcpu or comes from Lite timer
  ➢ Maskable overflow interrupt
  ➢ PWM signal generator (programmable duty cycle, polarity control, maskable compare interrupt, freq range 2kHz-4MHz at Fcpu=8MHz)
  ➢ Output compare function

• AT2 (lite2):
  ➢ 6 external pins
  ➢ 4 independent PWM outputs: BREAK pin (force a break condition on PWM outputs) programmable duty cycle and output mode, polarity control, maskable compare interrupt, freq range 2kHz-4MHz at Fcpu=8MHz)
  ➢ 12-bit Input Capture function (ATIC pin): rising and falling edge
  ➢ Maskable IC and overflow interrupts
  ➢ Output compare function
Overview (2/2)
12-bit ARTimers

- AT3: Dual Auto reload timer (lite3, lite1b)
  - 6 external pins
  - 1 or 2 12-bit upcounters!
  - 4 independent PWM outputs: BREAK pin (force a break condition on PWM outputs) programmable duty cycle and output mode, polarity control, maskable compare interrupt, freq range 2kHz-4MHz at Fcpu=8MHz), Dead time generation (for Half Bridge driving mode)
  - 12-bit Input Capture function (ATIC or LTIC pin): rising and falling edge, long range input capture feature (Lite timer clock feeds ART timer, signal on LTIC: 20-bit cascaded)
  - Maskable IC and overflow interrupts
  - Output Compare function
SPGA
Software Programmable Gain Amplifier

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
Software Programmable Gain Amplifier

OVERVIEW

• Integrated RAIL to RAIL OpAmp

• Internal low programmable Gain (Up to 16)

• Integrated reference voltage sources, VCC dependent & independent (Band-Gap).

• OpAmp Outputs internally connected to ADC input

• Interrupt flag in comparator mode

• Power on/off bit & active in low power modes

• DAC capability with PWM/ART output
SPGA Block Diagram

Reference voltages:
* 1.2V, Vcc independant
* 8 steps, VCC dependant

Analog (Amplifier) or digital (Comparator) output
SPGA MODES (1)

Programmable gain Value

<table>
<thead>
<tr>
<th>Inverter</th>
<th>-1</th>
<th>-2</th>
<th>-3</th>
<th>-4</th>
<th>-8</th>
<th>-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non Inverter</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>16</td>
</tr>
</tbody>
</table>
OPAMP MODES (2)

Comparator mode

- OA1PIN To ADC Channel 8
- NS1[2:0] bits
- AZ1 bit
- G1[2:0] bits
- VR1E, PS1[1:0] bits
- VR1[2:0] bits
- bit OA1IE bit
- OA1 Interrupt
- selectables
- positive input

Band Gap Reference Voltage (1.2V)

VR1[2:0] bits x VDDA /8

8-Step Reference Voltage 1

AV CL = 1, 2, 4, 8, 16, ∞

R = 2K

OA1NIN

AGND

OA1O
SPGA MODES (3)

8-Bit Digital to Analog Converter
## Op-Amps characteristics comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>UA 741 general purpose</th>
<th>ST72C171</th>
<th>TS922 audio performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rail to Rail</td>
<td>no</td>
<td>yes (I/O)</td>
<td>yes (I/O)</td>
</tr>
<tr>
<td>Consumption (mA)</td>
<td>1.7</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>Output current (mA)</td>
<td>25</td>
<td>5&lt; Icc &lt; 50</td>
<td>80</td>
</tr>
<tr>
<td>GPB</td>
<td>1</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>Noise (nv*sqr(Hz))</td>
<td>23</td>
<td>&lt;50</td>
<td>4</td>
</tr>
<tr>
<td>Offset voltage (mV)</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Slow rate (v/us)</td>
<td>0.5</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>Power supply range (v)</td>
<td>4 to 44</td>
<td>3 to 6</td>
<td>2.7 to 10</td>
</tr>
</tbody>
</table>
ST7 CAN

- Optional features:
  - EEPROM
  - AD converter
  - Analog Comparator
  - SCI
  - SPI
  - I2C
  - 16-bit Timer
  - Lite Timers
  - 8-bit Auto Reload Timer
  - 12_bit ARtimers
  - Programmable OpAmp
  - CAN
High End Car Networks

ENTERTAINMENT
- Radio
- Audio
- Power
- TV-Module
- Very high speed up to 100M bps

BODY
- Door Module
- Door Module
- Trunk Module
- Air Cond
- Doom-Control
- Contr. Pannel.
- Light C. Front
- Light C. Front
- Door Module
- Door Module
- Door Module
- Driver-Seat
- Pass.-Seat
- Backseat
- Central Airbag
- Light C. Rear
- Light C. Rear
- Dashboard CAN Gateway

ACCESS
- Car-Phone
- Navigation
- Video-Box
- Wiper Left
- UART Node to Node

POWER-TRAIN
- Sensor
- EMS
- Gearbox
- Traction
- ABS

TRACTION STABILITY
- Low speed CAN 2.0B < 125 kbps
- High speed CAN2.0B 125 kbps to 1Mbps

STANDARDS
- ISO9141 10kbps
CAN High Lights

• Asynchronous Serial Communication Protocol Based On
  ➢ Multi-master concept
    ✓ CSMA/CA (Carrier Sense Multiple Access Collision Avoid)
    ✓ Message priority
  ➢ Object oriented communication
    ✓ No node addressing, but content identification
  ➢ Realtime capability
    ✓ Low message transfer latency
  ➢ System wide message consistence
    ✓ Error detection & management mechanism
CAN Bus Topology

- CANH
- CANL
- CAN Transceiver
- CAN Controller
- \( \mu \)C
- Sensor Signals
- Actuator Signals
- Node 1
- Node 2
- Node N
- differential voltage
- twisted wire
- 40m @ 1 Mbps
Standard vs Extended Frames

CAN 2.0A (standard format: 11 bit identifier)

CAN 2.0B (extended format: 29 bit identifier)
## Standard vs Extended Implementations

<table>
<thead>
<tr>
<th>Implementation</th>
<th>2.0 A</th>
<th>2.0B Passive</th>
<th>2.0B Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard send / receive</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Extended check / acknowledge</td>
<td>no (destructive !)</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Extended send / receive</td>
<td>no (destructive !)</td>
<td>no (compliant)</td>
<td>yes</td>
</tr>
</tbody>
</table>

2.0B passive and active nodes can cohabit on the same network
CAN Bus Characteristics

• Wired-AND
  - Two Levels
    - Recessive
    - Dominant

• Dominant Prevail
Bus Access And Arbitration

Node X
Node 1
Node 2
Node N
CAN Bus

Arbitration Phase
Remainder
Transmit Request

Node 1
Node 2
Node N
CAN Bus
geressive dominant
Error Detection - At Transmitter

- **Bit Error**
  - Based on the "write verify" principle each bit sent is read back by the transmitter
    - If the values differ an error is detected
    - Except during arbitration phase

- **Acknowledge Error**
  - A receiver considering a frame as correct has to acknowledge it otherwise it has to signal the detected error with an Error Frame
  - A transmitter missing the acknowledge supposes to be the only active node in the network
Error Detection - At Receiver

- Bit Error
  - See transmitter

- Stuff Error
  - More than 5 identical bits are detected

- CRC Error
  - The received and calculated CRC values different

- Form Error
  - One of the fixed format bits is not correct
    - CRC delimiter
    - ACK delimiter
    - End Of Frame
Fault Confinement

- **Error Active**
  - TEC > 127 or REC > 127
  - TEC < 128 and REC < 128
  - TEC > 255

- **Error Passive**
  - 128 blocks of 11 successive recessive bits

- **Bus Off**
  - Reset and configuration
Basic vs Full CAN

- Full CAN controller
  - Many buffers (e.g. 15)
  - Sophisticated message filtering
  - Dedicated HW
  - Larger silicon (buffer size, filtering)
  
  Typical: Car body

- Basic CAN controller
  - Few Tx/Rx buffers (e.g. 3)
  - Simple identifier filtering
  - More SW overhead
  - Less silicon
  
  Typical: Engine management
Filters Possible Implementations

- **FULL CAN**
  - Buffer allocation is static based on message ID
  - Messages may not be copied to RAM (automatic reply capability)
  - Interesting if great number of buffers

- **BASIC CAN**
  - Buffer allocation is dynamic based on availability
  - Messages must be copied to RAM (more CPU load)
ST7 CAN Cell - Block Diagram

- Internal Bus
  - Interface
  - TX / RX Buffer 1 (10 Bytes)
  - TX / RX Buffer 2 (10 Bytes)
  - TX / RX Buffer 3 (10 Bytes)
  - ID Filter 0 (4 Bytes)
  - ID Filter 1 (4 Bytes)
  - BCDL (8-bit Shift Reg.)
  - EML
  - CRC
  - CAN 2.0 B passive Engine

- Other Components:
  - PAGE
  - CKDIV
  - BTR
  - IT CTRL
  - CSR
  - TEC
  - REC
  - IT STAT

- RX and TX signals connected to respective components.

- Direction arrows indicating data flow.
Key Features

• Comply with 2.0B passive specification

• Speed: up to 1MBits/s

• Basic CAN capability with:
  ➢ 3 prioritized object messages (8 data bytes each)
  ➢ 2 acceptance filters (ID + RTR match for 0, 1 or don't care)

• Flexibility

• Full baud rate & bit timing control
  ➢ Buffers useable as transmit or receive buffers
Other Features

• Optimized buffer flip-flopping capability in transmission
  ➢ Real-time performance at minimum CPU load
  ➢ Deterministic transfers
  ➢ Solves the inner priority inversion problem with no extra hardware

• Low power mode

• Extensive interrupt scheme
  ➢ Separate signalling for successful transfers and errors
  ➢ 7 maskable sources: Tx, Rx123, Tx error, Rx error, overrun
Buffer Pagination

- Saves on address space
  - 44 registers mapped onto 22 addresses

- Allows more efficient addressing mode
  - No memory indirection

5 PAGES OF 16 REGISTERS
FOR DIAGNOSIS, MESSAGE BUFFERS, ACCEPTANCE FILTERS
Page Mapping

<table>
<thead>
<tr>
<th>Page 0</th>
<th>Page 1</th>
<th>Page 2</th>
<th>Page 3</th>
<th>Page 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAST ID</td>
<td>10-BYTE BUFFER 1</td>
<td>10-BYTE BUFFER 2</td>
<td>10-BYTE BUFFER 3</td>
<td>FILTER 0</td>
</tr>
<tr>
<td>RESERVED</td>
<td>RESERVED</td>
<td>RESERVED</td>
<td>RESERVED</td>
<td>FILTER 1</td>
</tr>
<tr>
<td>TEST REG.</td>
<td>CTRL / STATUS 1</td>
<td>CTRL / STATUS 2</td>
<td>CTRL / STATUS 3</td>
<td>RESERVED</td>
</tr>
<tr>
<td>ERROR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COUNTERS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Buffer-dedicated Control / Status Register (BCSR)
  - Reduces requirements on CPU reactivity
- Dual mapping of Rx interrupt flags in the ISR and BCSR
  - The same code handles any buffer
Sleep Mode & Wake Up

- Software controlled low power mode (RUN bit cleared)
  - Prescaler and protocol engine stopped
  - Need for resynchronization

- Automatic low power mode after 20 recessive bits
  - Protocol engine stopped
  - No need for resynchronization (idle mode)

- Wake-up upon reception of dominant bit

- Possible wake up pulse generation when leaving stand-by
Cell Certification

• Validated with Bosch C model and patterns

• Certification on going with expert third party (c&s)
  ➢ Protocol compliance test according to Bosch & ISO specifications
  ➢ CPU interface test
  ➢ Cell robustness behaviour

• Final silicon certification targeted 1Q99
ST725xx Kits

• Development kit - ST7MDT2-DVP2
  ➢ Available now at ST
  ➢ User software development and debugging
  ➢ Easy connection to a high speed CAN node/network
  ➢ Compatible to widespread CAN Vector tools

• CAN Demonstration Board
  ➢ Available now at ST
  ➢ Software functions controlled by PC - Visual CAN

• Visual CAN
  ➢ Intuitive GUI to discover and exercise ST7 passive CAN cell
  ➢ Existing CAN network traffic monitoring
  ➢ CAN messages generator function