ECSE 426
Board and HW Lab

Zeljko Zilic

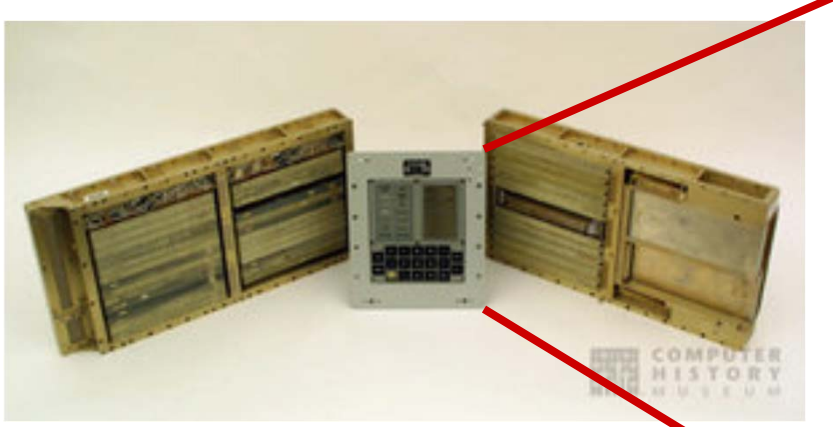
Room 536
McConnell Building
zeljko@ece.mcgill.ca
www.macs.ece.mcgill.ca/~zeljko

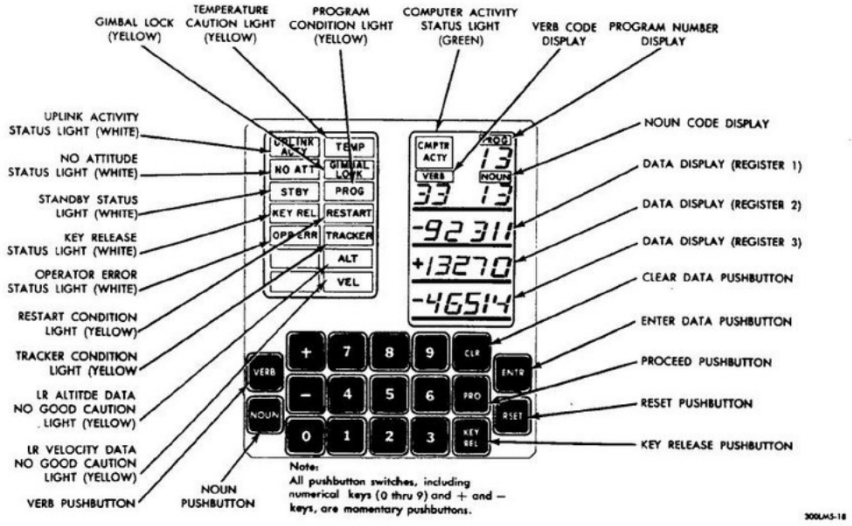
McGill

McGill

Quiz-time History Flashback

- Quote from 1st modern embedded system: Apollo Guidance Computer (AGC)





DSKY: Display/Keyboard

1201 alarm ("Executive overflow - no vacant areas")

1202 alarm ("Executive overflow - no core sets")

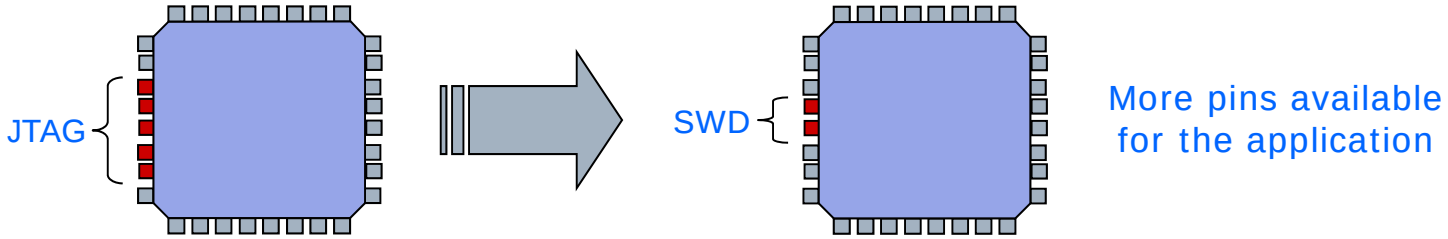
(quote from the first descent on Moon)

Topics for Today

- HW Interfacing: SPI and I^2C
- Peripherals: accelerometer with applications
- Exceptions with Cortex M processsors (time allowing)
- Lab 2 definition and some background

Cortex M Debug Capabilities

- Serial Wire Debugging for reduced device pin-out



- Embedded break/watch for easy flash application debugging
  - ♦ 8 hardware breakpoints, 2 hardware watchpoints
- Serial Wire Viewer for low bandwidth instruction and data trace
  - ♦ Powerful debugging peripheral, no extra hardware / software
  - ♦ Uses serial wire interface
  - ♦ View and modify variables and peripherals in real time!
  - ♦ ITM viewer for printf()-style debugging
  - ♦ Trace module counters for # of CPU clocks, interrupts, many more
- Debugging features still available in low power mode

McGill

Serial Wire Viewer Debug Features

- Data Read and Data Write tracing...*in real time !*
- ETM trigger
- Program Counter (PC) or Data address sampler registers
- ITM Viewer: printf-style debugging
- Various counters for:
  - # of CPU clock cycles
  - Total cycles spent in interrupt (exception) processing
  - # of cycles processor is sleeping
  - Total cycles spent in load/store operations.
  - # of instruction cycles

STM32F4xx Block Diagram

Cortex-M4 w/ FPU, MPU and ETM

Memory

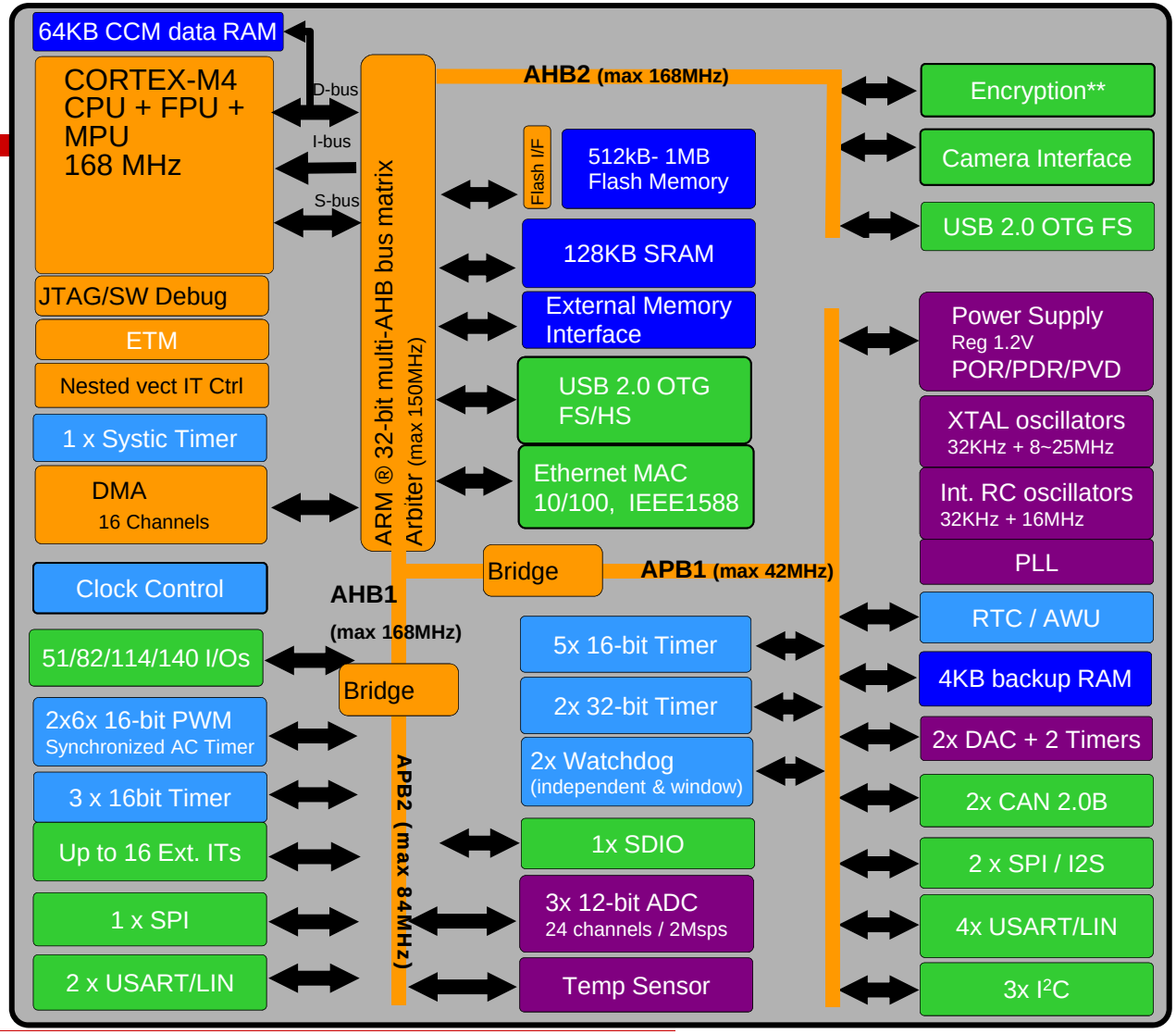
- Up to 1MB Flash memory
- 192KB RAM ->64KB CCM data RAM
- FSMC up to 60MHz

New application specific peripherals

- USB OTG HS w/ ULPI interface
- Camera interface
- HW Encryption**: DES, 3DES, AES 256-bit, SHA-1 hash, RNG.

Enhanced peripherals

- USB OTG Full speed
- ADC: 0.416µs conversion/2.4Msps, up to 7.2Msps in interleaved triple mode
- ADC/DAC working down to 1.8V
- Dedicated PLL for I2S precision
- Ethernet w/ HW IEEE1588 v2.0
- 32-bit RTC with calendar
- 4KB backup SRAM in VBAT domain
- Pure 1% RC
- 2 x 32bit and 8 x 16bit Timers
- high speed USART up to 10.5Mb/s
- high speed SPI up to 37.5Mb/s



,

** Encryption only on STM32F415 and STM32F417

McGill

Using F4-Discovery: Library

- STM32 library by ST Microelectronics
- Source and header files for peripherals
  - e.g., stm32f4xx_gpio.c, stm32f4xx_gpio.h
- To use libraries:
  - #include “stm32fxx.h”
  - Add source files (e.g., stm32f4xx_gpio.c) to project
  - Un-comment lines in stm32f4xx_conf.h invoking peripherals needed in project: #include “stm32f4xx_gpio.h”

STM32 Library – How to Use It?

- Function and constant for each peripheral has prefix with name, like: GPIO, TIM1:
  ie. **GPIO_Init**(), **ADC_Channel_0**, **USART_IT_TXE**
- Most of the settings is in **1fromN** convention and allow to use concatenation, like:
  GPIO_Pin_0 | GPIO_Pin_1 means: **pins 0 and 1** from will be configured in the same time
- There are predefined types in ***stm32f4xx.h*** file, like:
  - **u8** – unsigned char
  - **u16** – unsigned short
  - **RESET** / **SET**, **FALSE** / **TRUE**, **DISABLE** / **ENABLE**
- Most of the peripherals (PPP) has set of instruction:
  - **PPP_DeInit**(...) – set all PPP register to its reset state
  - **PPP_Init**(...) – validation of the configuration for the peripheral
  - **PPP_Cmd**(ENABLE/DISABLE) – turn on/off PPP peripheral (not affects its clock)
  - **PPP_ITConfig**(...) – configuration (on/off) of sources of interrupts for PPP peripheral
  - **PPP_GetFlagStatus**(...) – read flags from the peripheral (polling)
  - **PPP_ClearFlag**(...) – clear flags from the peripheral
  - **PPP_ClearITPendingBit**(...) – clear IRQ flag

McGill

GPIO Configuration - Basics

- After the **reset** all pins are in **input floating** mode
- Pins are grouped into 16bit ports (GPIOA, GPIOB, ... GPIOI)

- GPIO ports are configured by several registers [names follow reference manual] that are updated by **GPIO_Init()** function automatically with the values from the **GPIO_InitTypeDef** structure:
  - **GPIO_Pin** -> GPIO_Pin_0 .... 15, GPIO_Pin_All, GPIO_Pin_None
  - **GPIO_Mode**:
    - GPIO_Mode_AN //analog mode
    - GPIO_Mode_IN //input mode
    - GPIO_Mode_OUT //output mode
    - GPIO_Mode_AF //alternate function mode
  - **GPIO_OType**:
    - *GPIO_OType_PP*
    - *GPIO_OType_OD*
  - **GPIO_Speed**:
    - GPIO_Speed_2MHz //lowest EMI -> softer edges
    - GPIO_Speed_25MHz
    - GPIO_Speed_50MHz
    - GPIO_Speed_100MHz //highest EMI -> sharper edges
  - **GPIO_PuPd**:
    - *GPIO_PuPd_NOPULL*
    - *GPIO_PuPd_UP*
    - *GPIO_PuPd_DOWN*

McGill

GPIO Configuration - Task

Can have **LED_Config()** function (***main.c*** file) to configure lines 12..15 from port GPIOD:

- In GPIO state (used in LED_Blink and LED_Circle states):

  As general purpose output pins in push-pull configuration with 25MHz speed, without pull-up

Need the connection of the clock of used peripherals BEFORE the configuration

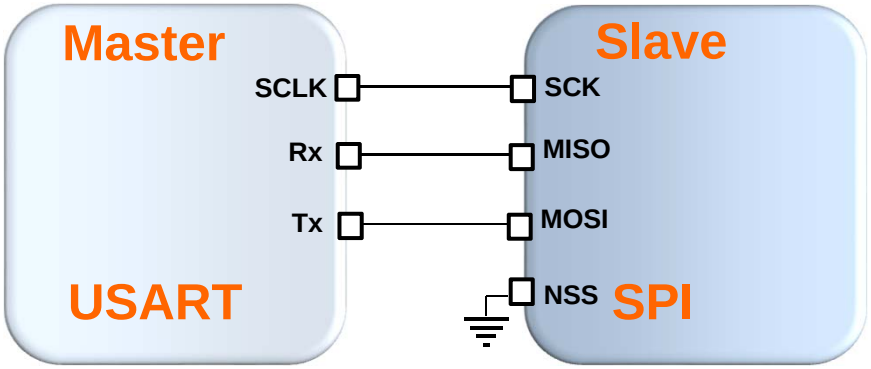
McGill

Synchronous Transmission

- Data clocked along with a clock signal
- Source and destination clocks not necessarily in lockstep
- Sender (master) controls the transmission clock
  - No data transmitted without a clock
  - Recipient (slave) has no control over clock
- *Bidirectional exchange* of data

Synchronous Mode - SPI

- USART for Full duplex synchronous communication
  - Full-duplex, three-wire synchronous transfer
  - USART Master mode only
  - Programmable clock polarity (CPOL) and phase (CPHA)
  - Programmable Last Bit Clock generation
  - Transmitter Clock output (SCLK)



Full Duplex

McGill

SPI Features With F4 Processors

- Two SPIs: SPI1 on high speed APB2 and SPI2 on low speed APB1
- Full duplex synchronous transfers on 3 lines
- Simplex synchronous transfers on 2 lines with or without a bi-directional data line
- Programmable data frame size :8- or 16-bit frame format selection
- Programmable data order with MSB-first or LSB-first shifting
- Master or slave operation
- Programmable bit rate: **up to 37.5 MHz in Master/Slave mode**
- NSS management by hardware or software for both master and slave: Dynamic change of Master/Slave operations
- TI mode (master and slave operations).

McGill

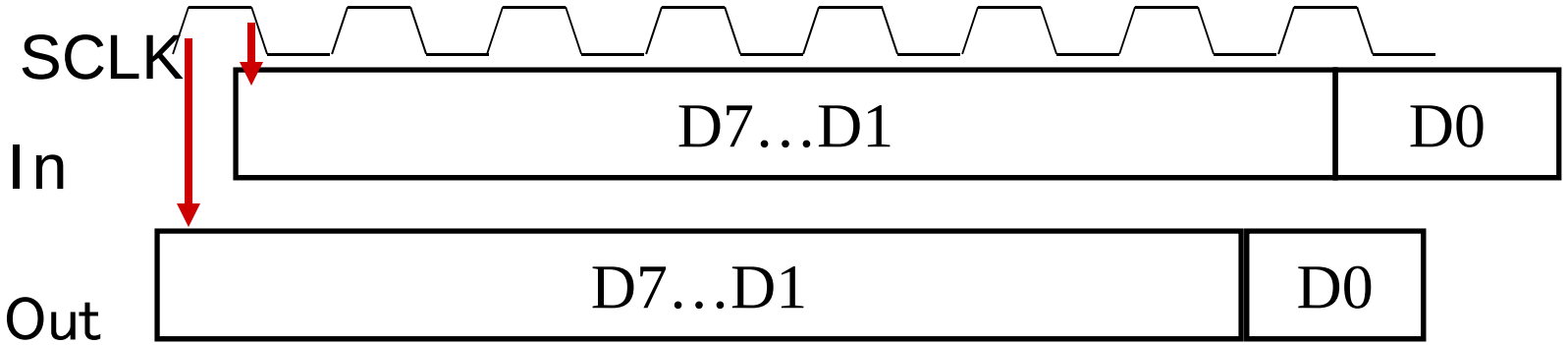
SPI Features (2/2)

- Programmable clock polarity and phase
- Dedicated transmission and reception flags (Tx buffer Empty and Rx buffer Not Empty) with interrupt capability
- SPI bus busy status flag
- Master mode fault and overrun flags with interrupt capability
- Hardware CRC feature for reliable communication
- Support for DMA

McGill

SPI Character Format

- Character transmission formats:



- D0-D6 (D7) : programmable to 7 or 8 data bits (or more)
- Programmable clock polarity, clock rate

SPI Signals

- 4 Wires in total, 3 could be used
  - SIMO: Slave in, master out
  - SOMI: Slave out, master in
  - SCLK: USART clock for SPI
  - SSN: Slave select (4-pin mode)
- Multiple-slave protocol possible through use of SSN
  - Select another device to transmit

SPI Circuitry in USART - Example

- Similar to UART
  - Two shift regs.
  - Flexible clocks
- SW use similar to UART
  - Init, set registers