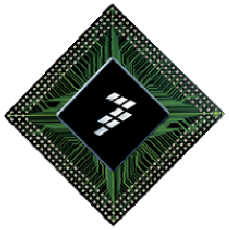
October 23, 2008

Quick Start
with the Self-Driven Slot Car Development

Freescale Race Challenge 2009



Milan Brejl

Freescale Semiconductor, Rožnov p.R., Czech Republic

freescale™ semiconductor

ČESKÁ AUTODRÁHA FARO CZECHIA RACING TRACK & SRC

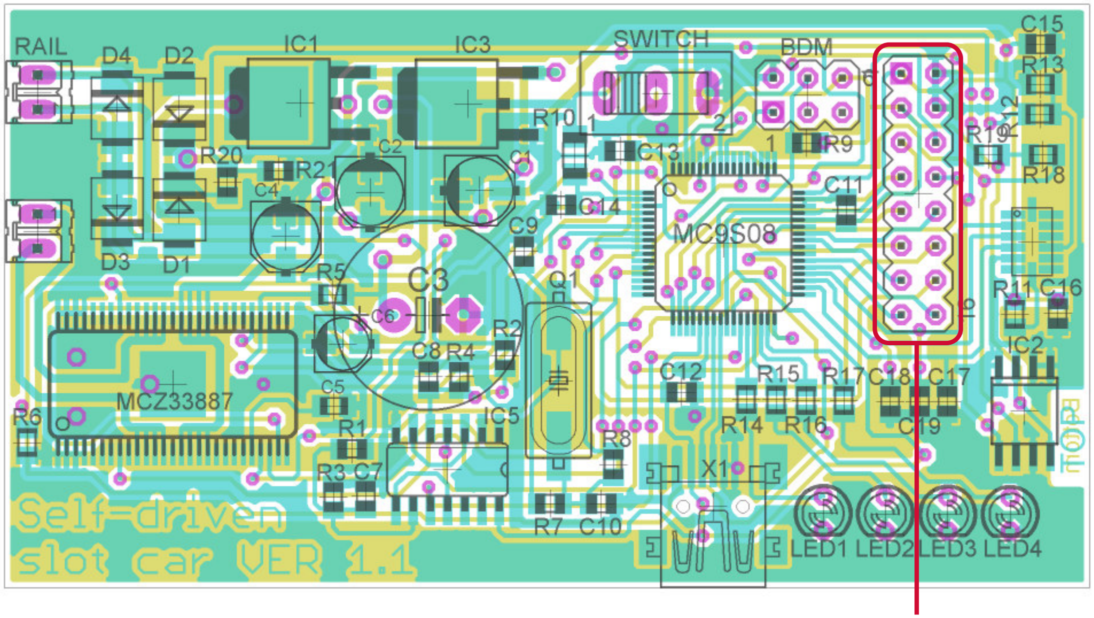
Freescale Race Challenge 2009: Quick Start

- Populate the PCB
- Build the electronics into the slot car
- Install development tools
- Open example application
- Program the flash memory
- Run the slot car on a test track
- Download the measured data from EEPROM to PC
  - Tip: Analyze the signal data in Matlab

- Develop your own self-driving algorithm

Populate PCB

- Schematic, PCB layout and Bill Of Materials (BOM) available:
  - slot_car_equip\HW\schematic
  - slot_car_equip\HW\PCB
  - slot_car_equip\HW\BOM.txt

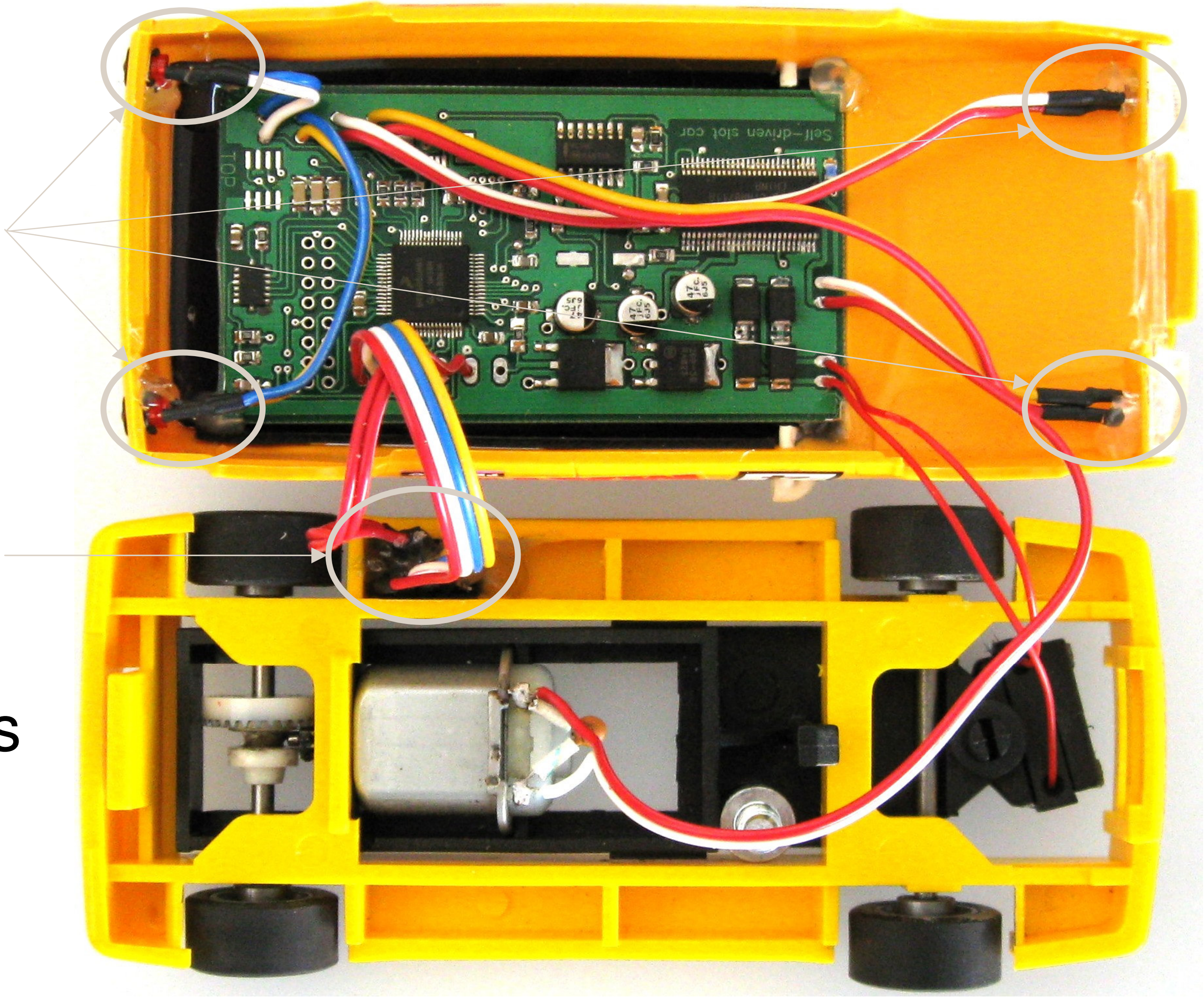


expansion connector

- Notes
  - You can design an expansion board and connect to the expansion connector available on the provided PCB.
  - You can exchange the S08JM 8-bit microcontroller with it’s 32-bit brother MCF51JM in the same package and the same pinout.
  - You can redesign the slot car electronics by your own.
    - feel free to start from the provided schematic, layout and Eagle libs.

Build Electronics Into Slot Car

- Tips:
  - Mount LEDs into head and break lights for run-time diagnostics
  - Mount the BDM debugging connector and the switch to chassis for external access



Install Development Tools

- Install Freescale CodeWarrior
  - install\CW_MCU_V6_2_SE.exe

- Install Freescale FreeMASTER
  - install\fmaster13-8.exe

Open Example Application

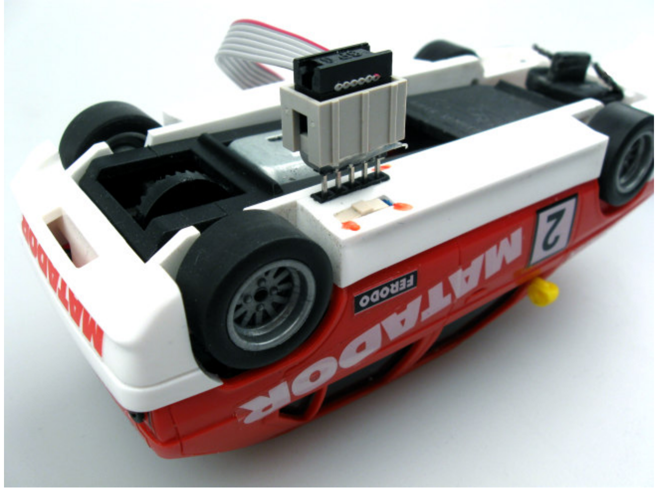
- ►The example application project is available at
  - slot_car_equip\SW
- ►Run slot_car_equip\SW\project.mcp CodeWarrior project

- ►The application code demonstrates how to:
  - initialize the S08JM32 microcontroller
  - use the motor H-bridge driver and drive the car
  - use the accelerometer and sample X, Y, and Z forces
  - sample motor current and DC-bus voltage
  - use the EEPROM and store/load measured data
  - use the LEDs and read the switch

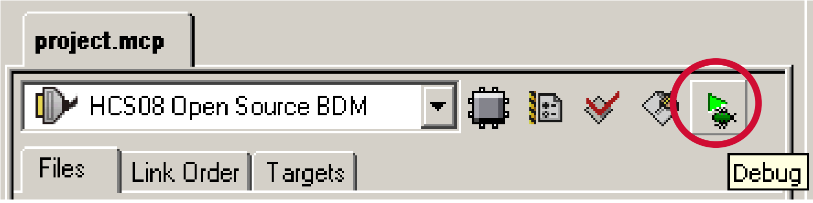
Program Flash Memory

- ►Connect the OpenSourceBDM debugger to USB
- ►If prompted, install the OpenSourceBDM Windows driver
  - install\OpenSourceBDMDriver\OpenSourceBDM.inf

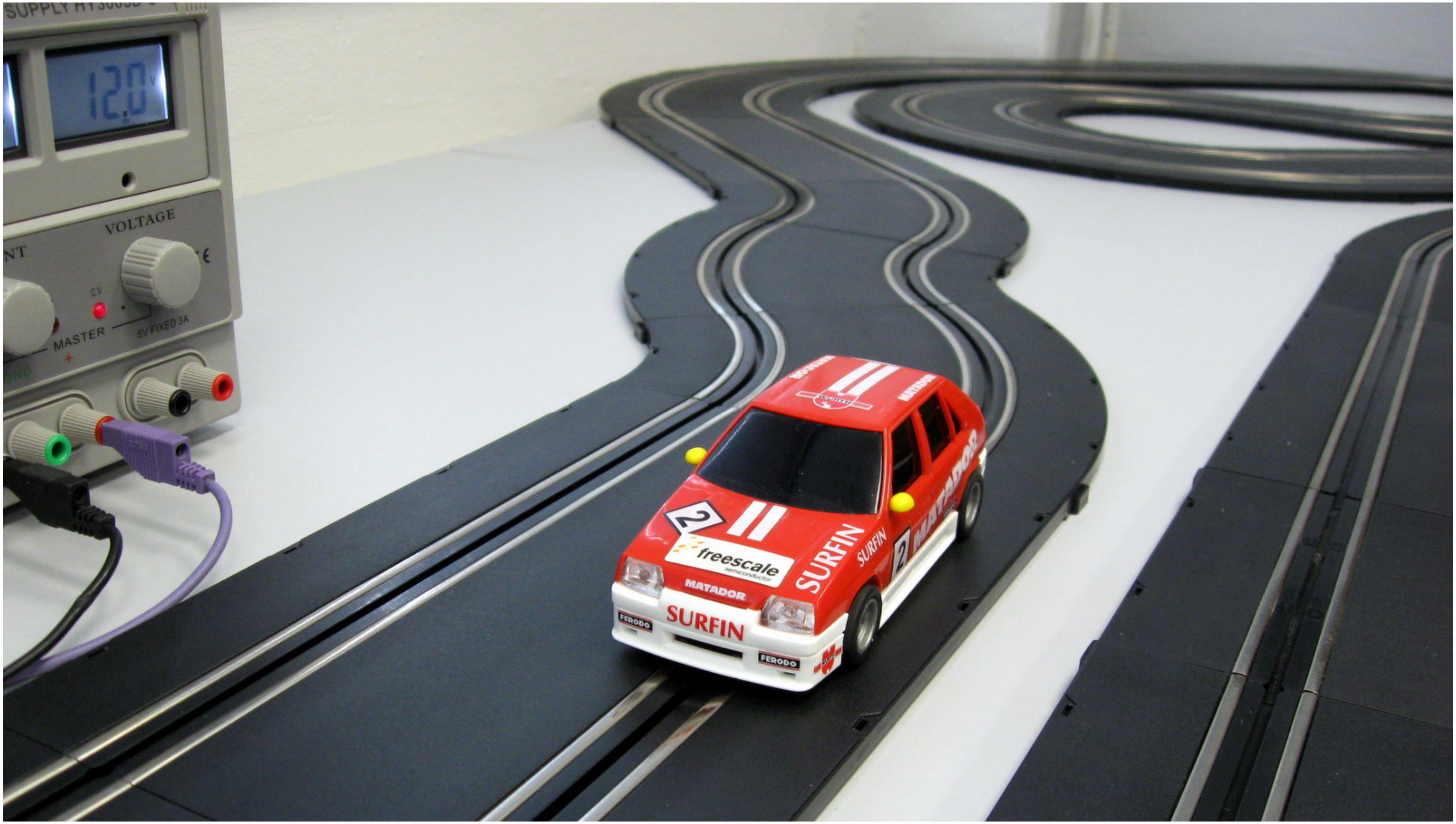
- ►Connect the debugger to the slot car BDM port



- ►Push Debug button in the CoderWarrior
  - confirm BDM connection
  - confirm flash re-writing



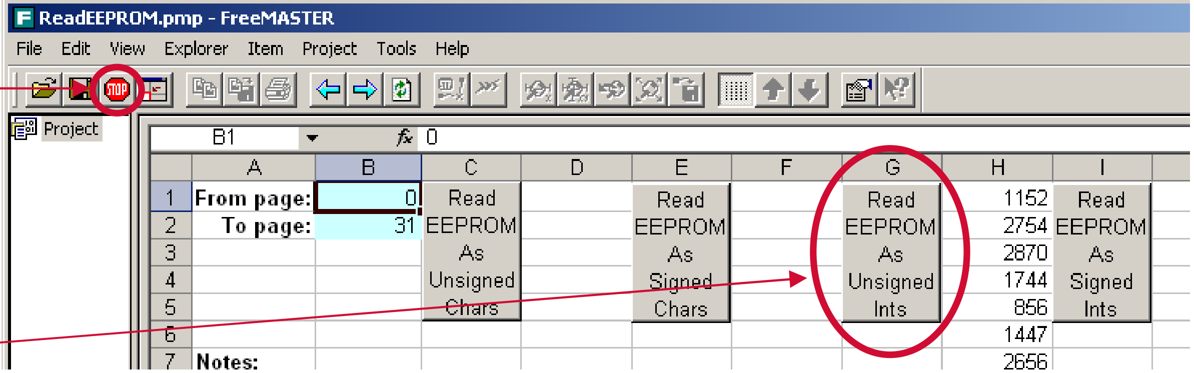
Run Slot Car on Test Track



Download Measured Data from EEPROM to PC

- ►Close CodeWarrior
- ►Connect the OpenSourceBDM debugger to the slot car

- ►Run ReadEEPROM.pmp FreeMASTER project
  - slot_car_equip\SW\ReadEEPROM.pmp
- ►Ensure the FreeMASTER communication with the car is established (STOP button is up)
- ►Push button Read ... in the integrated Excel application

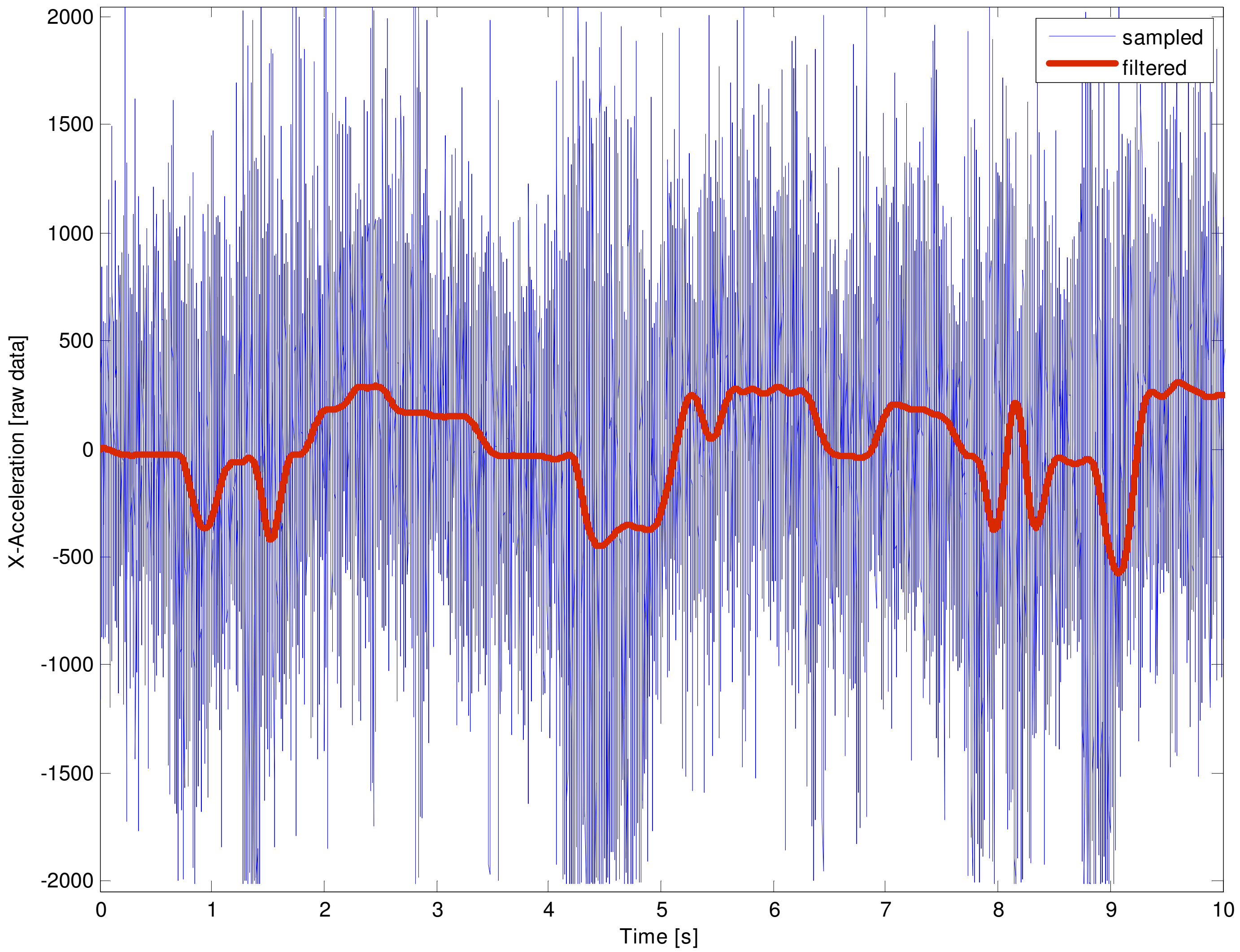


B1 ▾ fx 0

	A	B	C	D	E	F	G	H	I
1	From page:	0	Read EEPROM As Unsigned Chars		Read EEPROM As Signed Chars		Read EEPROM As Unsigned Ints	1152	Read EEPROM As Signed Ints
2	To page:	31						2754	
3								2870	
4								1744	
5								856	
6								1447	
7	Notes:							2656	

EEPROM data read

Tip: Analyze The Signal Data in Matlab



Need more computational power?

- ►The provided microcontroller S08JM32
  - Belongs to the Controller Continuum family called Flexis
  - Has a Flexis “brother” MCF51JM with
    - ColdFire V1 32-bit core and more memory,
    - the same peripherals, the same package and the same pinout
- ►If your self-driving algorithm is getting to the edge of the S08JM power (only the most sophisticated algorithms could), exchange it with MCF51JM

Learn more about the Flexis family: http://www.freescale.com/webapp/sps/site/overview.jsp?nodeId=016246233A3B62

Develop Your Own Self-Driving Algorithm

Good Luck!

freescale™
semiconductor