INDUSTRIAL 40 NM DEMONSTRATOR NUCLEO-STM32H755ZI-Q

Jiři Kadlec, Lukáš Kohout
kadlec@utia.cas.cz kohoutl@utia.cas.cz

Revision history

Rev.	Date	Author	Description
0	1.12.2019	J. Kadlec	Initial release, STM32H755ZI-Q benchmarks
1	12.05.2020	J. Kadlec	Added references
2	31.05.2020	J. Kadlec	Added STM32F767ZI, STM32HA3ZI-Q, STM32F743ZI, STM32H53ZI,
3	27.7.2021	J. Kadlec	Added STM32H723ZG (520 MHz), Zynq Ultrascale+ (1200 MHz)

WAKeMeUP

WAKeMeUP has been accepted for funding within the Electronic Components and Systems For European Leadership Joint Undertaking in collaboration with the European Union's H2020 Framework Programme (H2020/2014-2020) and National/Local Authorities, under grant agreement n° 783176

ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership

Table of Contents

1 Industrial 40 nm UTIA demonstrator........1

1.1. STM NUCLEO-H755ZI board........2

1.2. AdaFruit 1.8" TFT display shield V2........3

1.3. AdaFruit 1.8" TFT display shield V1........4

1.4. ArduZynq TE0723-03M board........5

2 Benchmarks performed on the demonstrator........8

2.1. Adaptive identification algorithms in Scilab........8

2.2. Steps to run menu-driven Scilab demo on terminal:........9

2.3. C interface for benchmarks in Scilab........11

2.4. C interface for benchmarks on STM32H755ZI-Q........12

2.5. Performance on ArduZynq and STM32H755ZI-Q........13

2.6. Projects released for STM32H755ZI-Q board........16

2.7. Performance of STM32 F767, H7A3, H743/53, H723........19

2.8. Released projects for STM32H743........20

2.9. Released projects for STM32H753........21

2.10. Released projects for STM32F767........22

2.11. Released projects for STM32H7A3........23

2.12. Released projects for STM32H723ZG........24

3 Conclusion........26

References........27

Disclaimer........28

Table of Figures

Figure 1: NUCLEO-H755ZI terminal with menu control of Scilab algorithms on ArduZynq........1

Figure 2: STMicroelectronics NUCLEO-H755ZI board........2

Figure 3: Adafruit TFT shield used data pins (left), used SPI serial communication pins........3

Figure 4: Adafruit TFT buttons and joystick (left), used I2C communication pins (right)........3

Figure 5: Pins used by AdaFruit TFT shield V1 with I2C expander chip........4

Figure 6: Pins used by "clasic" AdaFruit TFT shield V1........4

Figure 7: Trenz Electronic ArduZynq shield........5

Figure 8: Serial communication based connection of STM753ZI CM7 MCU and ArduZynq........6

Figure 9: NUCLEO-H753ZI connectors........7

Figure 10: Selection of identification algorithm as C MEX on ArduZynq Scilab.........8

Figure 11: Menu (left) serves for selection of DSP algorithms in Scilab on ArduZynq (right).........8

Figure 12: Result of adaptive identification benchmark.........10

Figure 13: DP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM7........13

Figure 14: SP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM7........13

Figure 15: SP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM4........14

Figure 16: DP FP MFLOP/s: RLS QRD identification on ArduZynq and CM7 14
Figure 17: SP FP MFLOP/s: RLS QRD identification on ArduZynq and CM7 15
Figure 18: SP FP MFLOP/s: RLS QRD identification on ArduZynq and CM4 15
Figure 19: DP FP MFLOP/s: RLS QR identificationon ArduZynq and CM7...........................15
Figure 20: SP FP MFLOP/s: RLS QR identification on ArduZynq and CM7..........................16
Figure 21: SP FP MFLOP/s: RLS QR identification on ArduZynq and CM4..........................16
Figure 22: Inverse QRD identification projects, STM32H755ZI-Q, CM7, CM4 MCUs 17
Figure 23: QRD identification projects (without sqrt), STM32H755ZI-Q, CM7, CM4 MCUs...17
Figure 24: QR identification projects (with sqrt) STM32H755ZI-Q, CM7, CM4 MCUs...........18
Figure 25: Combinations of boards and displays supported by released projects.................18
Figure 26: SP MFLOP/s: Comparison of A53, A9 MPUs with F767, H7A3, H743. H753, H723 MCUs. 19
Figure 27: DP MFLOP/s: Comparison of A53, A9 MPUs with F767, H7A3, H743, H743, H753, H723 MCUs. 19
Figure 28: Inverse QRD identification projects, STM32H743ZI2, CM7 MCU 400 MHz 20
Figure 29: Combinations of boards and displays supported by released projects.................20
Figure 30: Inverse QRD identification projects, STM32H753ZI, CM7 MCU 400 MHz 21
Figure 31: Combinations of boards and displays supported by released projects.................21
Figure 32: Inverse QRD identification projects, STM32F767ZI, CM7 MCU 216 MHz............22
Figure 33: Combination of board and display supported by released projects 22
Figure 34: Inverse QRD identification projects, STM32H7A3ZI-Q, CM7 MCU 280 MHz.......23
Figure 35: Combination of board and display supported by released projects 23
Figure 36: Inverse QRD identification projects, STM32H7A3ZI-Q, CM7 MCU 280 MHz.......24
Figure 37: Combination of board and display supported by released projects 24
Figure 38: Double precision system identification, STM32H755ZI-Q board, CM7 MCU........25

Acknowledgement

This work has been partially supported from project WakeMeUP, project number ECSEL 783176 and the corresponding Czech NFA (MSMT) institutional support project 8A18001.

1 INDUSTRIAL 40 NM UTIA DEMONSTRATOR

The terminal use the NUCLEO-H753ZI evaluation board produced by the ST Microelectronics with 400 MHz CM7 MCU, 2MB eFLASH and 1MB SRAM on single CMOS device.

NUCLEO-H753ZI board controls the Adaruit 1.8'' colour TFT display V2 with joystick. The resolution of the display is 160x128 pixels.

NUCLEO-H753ZI board supports serial communication with the ArduZynq shield.

The ArduZynq shield works with Xilinx 28 nm Zynq device XC7Z010-1C with dual-core Arm Cortex A9 MPU running at 650 MHz and 512 Mbyte DDR3 .

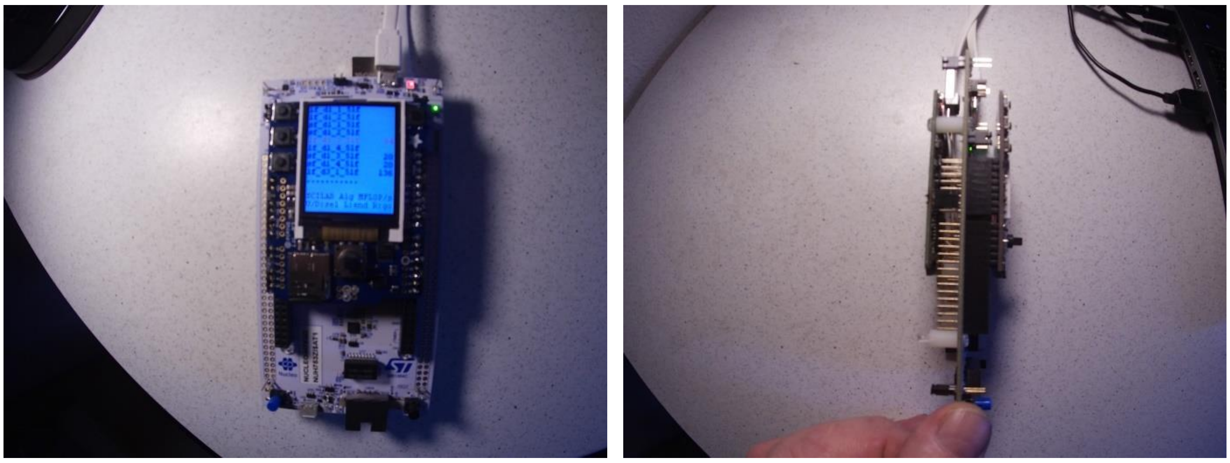

Figure 1: NUCLEO-H755ZI terminal with menu control of Scilab algorithms on ArduZynq

UTIA supports the Xilinx SDSoC 2018.2 compiler for the ArduZynq shield. The compiler serves for generation of HW accelerators with data movers (DMA or SG-DMA) from user defined C/C++ functions.

The programmable logic part of the Zynq device is configured with SIMD FP01x8 run-time reprogrammable single-precision floating point HW accelerator IP core.

The SIMD FP01x8 accelerator can be re-programmed in runtime by change of firmware. The firmware can be compiled directly on the A9 processor on the ArduZynq shield. Xilinx SDSoC 2018.2 compiler is not needed for this run-time re-compilation and reconfiguration of the firmware.

The dual core A9 processor on the ArduZynq shield is running Debian OS with pre-configured Petalinux 2018.2 kernel.

The ArduZynq shield is running Scilab command-line interpret client as Debian OS user space application. Scilab supports interpretation of double precision scripts and functions working with double precision matrix data.

The Scilab interpret can also call and execute compiled C functions written in Matlab MEX format. These functions can be compiled directly on the ArduZynq shield by C/C++ compiler, which is part of the Debian OS. The NUCLEO-H755ZI board serves as serial terminal with menu-based GUI for selection of Scilab demos to be executed in the ArduZynq shield.

1.1. STM NUCLEO-H755ZI board

The NUCLEO-H753ZI is the main PCB board. The STM753ZI CM7 MCU device is powered and programmed by micro USB cable from a PC. See Figure 2.
It is produced by STMicroelectronics. See:
https://www.st.com/en/evaluation-tools/nucleo-h753zi.html?ecmp=tt9470_gl_link_feb2019&rt=db&id=DB3171#resource

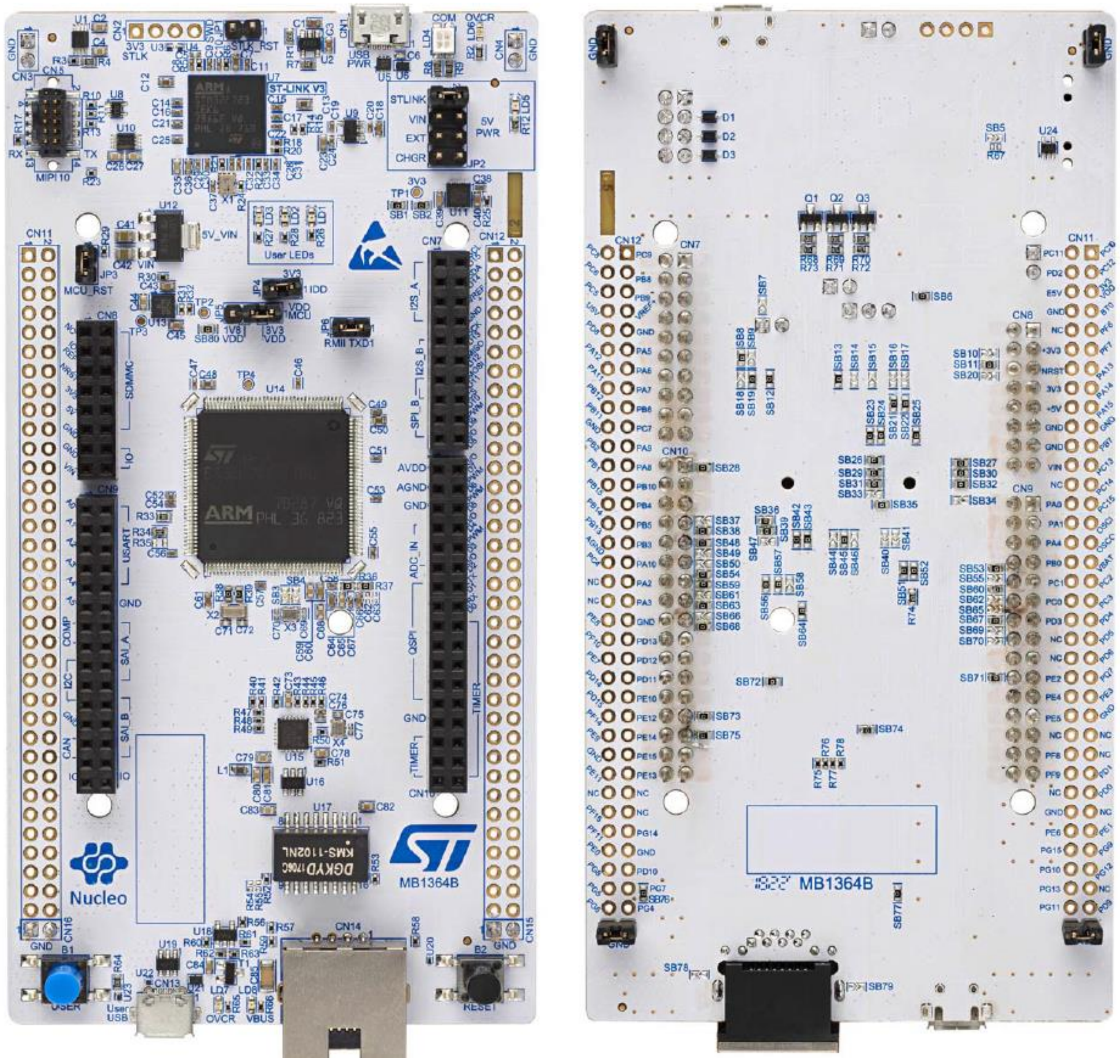

Figure 2: STMicroelectronics NUCLEO-H755ZI board

1.2. AdaFruit 1.8” TFT display shield V2

The AdaFruit 1.8” TFT display shield V2 with joystick and uSD card is connected to NUCLEO-H753ZI PCB by the Arduino connectors. This shield is produced by AdaFruit. See: https://learn.adafruit.com/1-8-tft-display/1-8-tft-shield

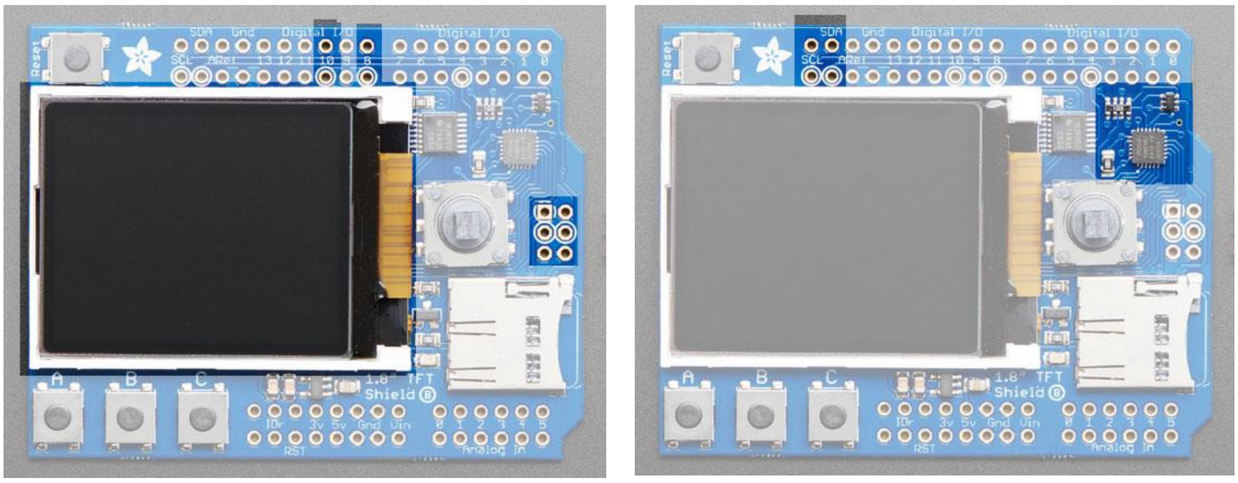

Figure 3: Adafruit TFT shield used data pins (left), used SPI serial communication pins

The 1.8" TFT display is full colour (16-bit RGB), 128x160 pixels, and has a backlight. The display receives data over SPI plus two pins:

- SCK - SPI Clock
- MOSI - SPI Data direction from STM753ZI CM7 MCU to AdaFruit shield
- Digital 10 - Chip Select
- Digital 8 - Data/Command Select

These 4 pins are controlled from the STM753ZI CM7 MCU.

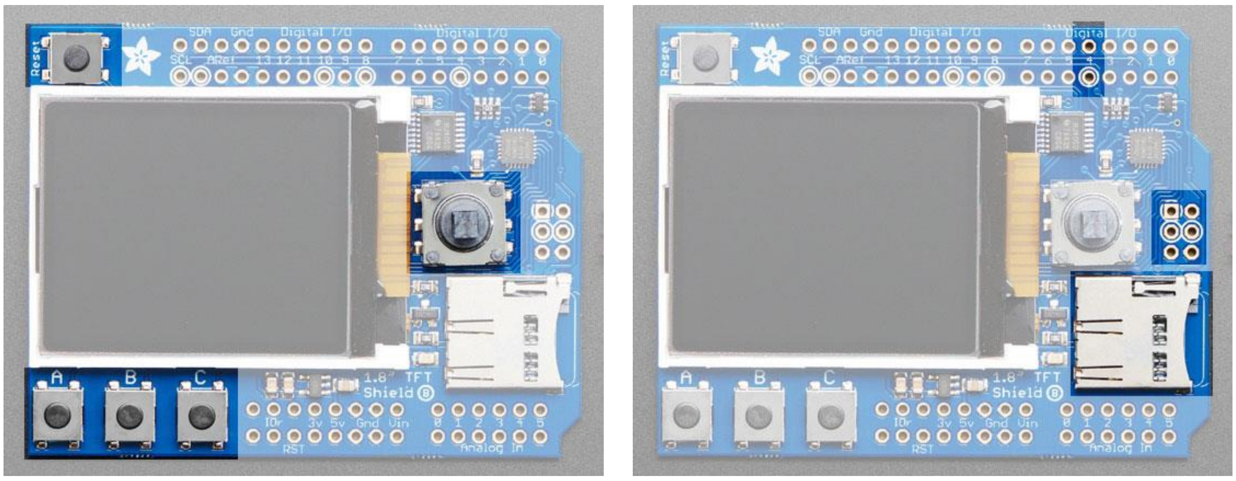

Figure 4: Adafruit TFT buttons and joystick (left), used I2C communication pins (right)

In the top left is the Reset button. It will reset the shield and the STM753ZI CM7 MCU when pressed. There are three buttons labelled A B C below the TFT, these are connected to the

I2C expander chip. To the right of the TFT is a 5-way joystick. It can be pushed up/down/left/right and select (in). You can read the joystick over I2C.

- SCL – I2C clock
- SDA – I2C data

The STM753ZI CM7 MCU can use the micro SD card slot to read/write data from any micro SD card. The SD card is connected to

- SCK - SPI Clock
- MOSI - SPI Data direction from STM753ZI CM7 MCU to AdaFruit shield
- MISO - SPI Data direction from to AdaFruit shield to STM753ZI CM7 MCU
- Digital 4 - Chip Select

Instead of taking several GPIO pins to read the buttons and joystick, as well as controlling the TFT backlight, an I2C expander chip is used. It is connected to the SDA/SCL pins and can read/write pins. The I2C expander is also used for control of the TFT backlight and reset. Connections are described in Figure 5 and Figure 9.

NUCLEO-H753ZI		**AdaFruit shield V2**	
D15 PB8	pin 2 in CN7 connector	SCL	SCL clock I2C expander chip
D14 PB9	pin 4 in CN7 connector	SDA	SDA data I2C expander chip
D13 PA5	pin 10 in CN7 connector	SCK	SCK SPI clock (wired to D13)
D12 PA6	pin 12 in CN7 connector	MISO	MISO SPI data (wired to D12)
D11 PB5	pin 14 in CN7 connector	MOSI	MOSI SPI data (wired to D11)
D10 PD14	pin 16 in CN7 connector	TFT_CS	TFT Chip Select
D8 PF3	pin 20 in CN7 connector	TFT_DC	Data/Command Select
D4 PE14	pin 8 in CN10 connector	SD_CS	SD card Chip Select
NRST RESET	Pin 5 in CN8 connector	RESET	RESET button
IOREF IOREF	Pin 3 in CN8 connector	IOREF	IOREF (VDD MCU) 3.3V
GND	Pin 11,13 CN8 connector	GND	GND
GND	Pin 8 in CN7 connector	GND	GND
+5V	Pin 9 in CN8 connector	+5V	+5V

Figure 5: Pins used by AdaFruit TFT shield V1 with I2C expander chip

1.3. AdaFruit 1.8" TFT display shield V1

If the "original" AdaFruit shield V1 (V1 is version without I2C expander chip) is used, the joystick position is identified by an analogue voltage value. I2C is not used. Connections are described in Figure 6 and Figure 9.

NUCLEO-H753ZI		**AdaFruit shield V2**	
A3 PB1	pin 7 in CN7 connector	A3	A3 analogue voltage value
D13 PA5	pin 10 in CN7 connector	SCK	SCK SPI clock (wired to D13)
D12 PA6	pin 12 in CN7 connector	MISO	MISO SPI data (wired to D12)
D11 PB5	pin 14 in CN7 connector	MOSI	MOSI SPI data (wired to D11)
D10 PD14	pin 16 in CN7 connector	TFT_CS	TFT Chip Select
D8 PF3	pin 20 in CN7 connector	TFT_DC	Data/Command Select
D4 PE14	pin 8 in CN10 connector	SD_CS	SD card Chip Select
NRST RESET	Pin 5 in CN8 connector	RESET	RESET button
GND	Pin 11,13 CN8 connector	GND	GND
GND	Pin 8 in CN7 connector	GND	GND
+5V	Pin 9 in CN8 connector	+5V	+5V

Figure 6: Pins used by "clasic" AdaFruit TFT shield V1

1.4. ArduZynq TE0723-03M board

The ArduZynq board is Arduino compatible Xilinx Zynq-7010 FPGA module with
Xilinx Zynq SoC XC7Z010-1CLG225C device, 512 MByte DDR3L,
Arduino Compatible connector, USB OTG, on-board USB JTAG and UART.

It is produced by by Trenz Electronic. See:
https://shop.trenz-electronic.de/en/Products/Trenz-Electronic/TE07XX-Zynq-SoC/TE0723-Zynq-SoC/

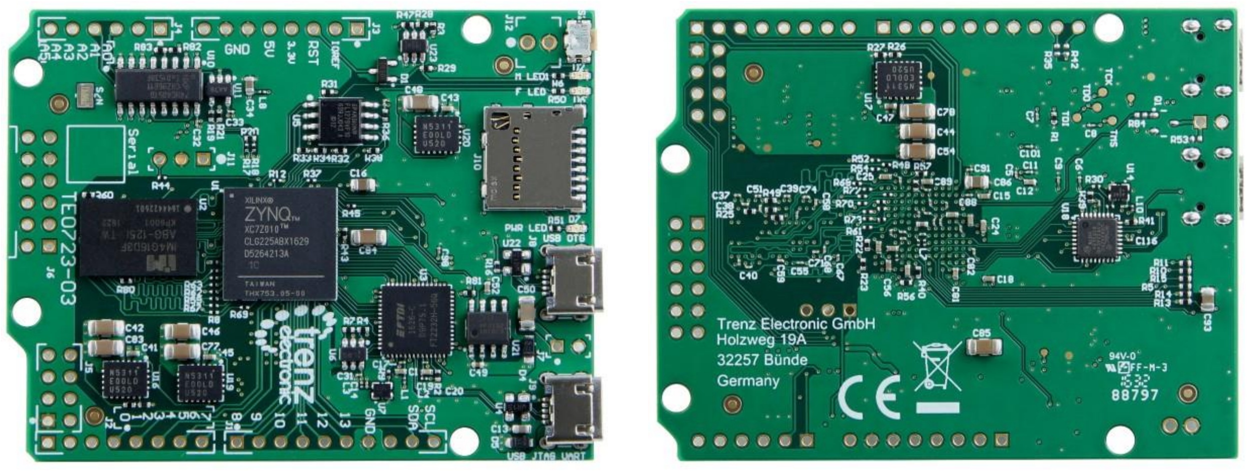

Figure 7: Trenz Electronic ArduZynq shield

Details about Xilinx Petalinux 2018.2 kernel, Debian file system configuration and UTIA support for the Xilinx SDSoC 2018.2 compiler can be found in application note "FP01x8 Accelerator on TE0723-03M".

http://sp.utia.cz/results/te0723_fp01x8/AppNote-te0723_fp01x8.pdf

and in the corresponding evaluation package published at

http://sp.utia.cz/index.php?ids=results&id=te0723_fp01x8 .

The Debian OS can use the Serial console linked via micro USB cable to PC. Same micro USB cable provides also the 5V power supply for the shield. Same micro USB cable can be used for download of BOOT.bin file containing the bit-stream, FSBL executable and u-boot executable and also for the In-circuit Logic Analyser (ILA).

The ArduZynq board is booting from the internal flash configured by the BOOT.bin file. In the next stage the U-boot running on A9 MPU loads the Petalinux 2018.2 image from the DOS-32 partition of the micro SD card. This partition has file system R/W accessible from Win 10.

The Petalinux image is complemented with Debian 9.8 "stretch" file system present in separate partition of the micro SD card.

The ArduZynq board is connected to the NUCLEO-H753ZI is the main PCB board and to the STM753ZI CM7 MCU device by simple two wire UART serial line with BaudRate 460800 bits/s. See Figure 8 and Figure 9.

NUCLEO-H753ZI		ArduZynq shield	
Tx PC6 D16	pin 1 in CN7 connector	Rx	pin 7 in J5 connector (wired to D16)
Rx PC7 D21	pin 11 in CN7 connector	Tx	pin 2 in J5 connector (wired to D21)
GND	Pin 11,13 CN8 connector	GND	GND
GND	Pin 8 in CN7 connector	GND	GND

Figure 8: Serial communication based connection of STM753ZI CM7 MCU and ArduZynq

This serial asynchronous communication is configured in the main application program of the STM753ZI CM7 MCU terminal application:

main.c

```
…
UartHandle.Instance = USARTx;
UartHandle.Init.BaudRate = 460800;
UartHandle.Init.WordLength = UART_WORDLENGTH_8B;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_NONE;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX;
UartHandle.Init.OverSampling = UART_OVERSAMPLING_16;
UartHandle.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
…
```

main.h

```
…
/* Definition for USARTx clock resources */
#define USARTx                           USART6
#define USARTx_CLK_ENABLE()              __HAL_RCC_USART6_CLK_ENABLE()
#define USARTx_RX_GPIO_CLK_ENABLE()      __HAL_RCC_GPIOC_CLK_ENABLE()
#define USARTx_TX_GPIO_CLK_ENABLE()      __HAL_RCC_GPIOC_CLK_ENABLE()
#define USARTx_FORCE_RESET()             __HAL_RCC_USART6_FORCE_RESET()
#define USARTx_RELEASE_RESET()           __HAL_RCC_USART6_RELEASE_RESET()
/* Definition for USARTx Pins */
#define USARTx_TX_PIN                    GPIO_PIN_6
#define USARTx_TX_GPIO_PORT              GPIOC
#define USARTx_TX_AF                     GPIO_AF7_USART6
#define USARTx_RX_PIN                    GPIO_PIN_7
#define USARTx_RX_GPIO_PORT              GPIOC
#define USARTx_RX_AF                     GPIO_AF7_USART6
…
```

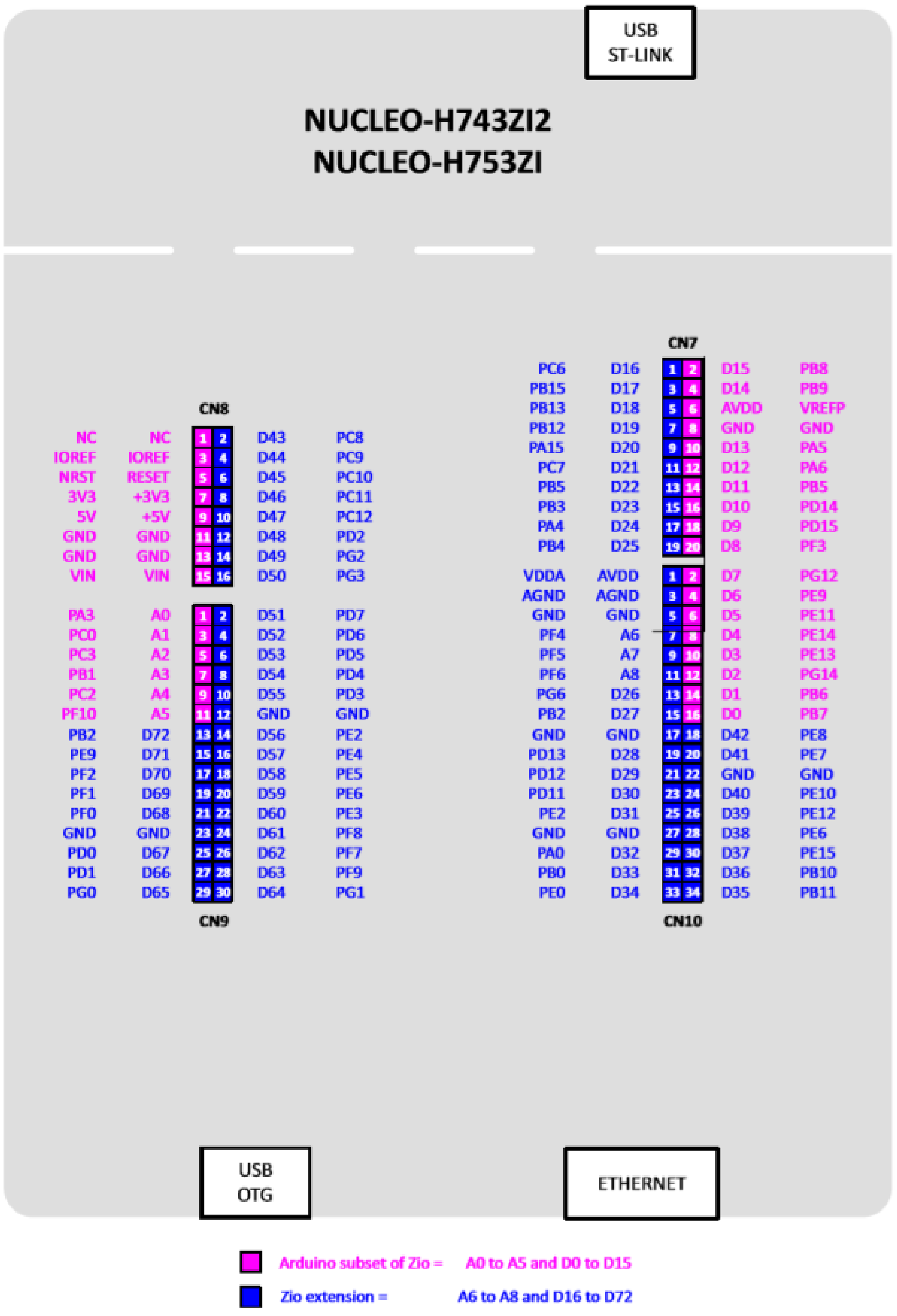

Figure 9: NUCLEO-H753ZI connectors

2 BENCHMARKS PERFORMED ON THE DEMONSTRATOR

The terminal is based on NUCLEO-H753ZI with AdaFruit TFT display shield. The ArduZynq shield serves for menu-based selection of adaptive identification algorithms [1] – [6] defined and executed in Scilab interpret running on the ArduZynq shield. See Figure 10.

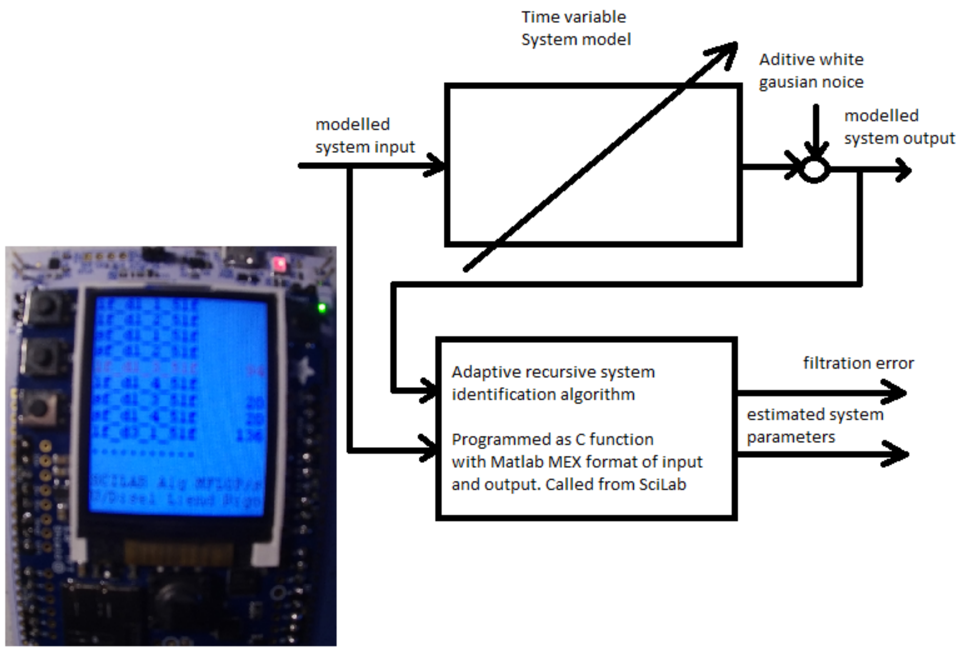

Figure 10: Selection of identification algorithm as C MEX on ArduZynq Scilab.

2.1. Adaptive identification algorithms in Scilab

The active menu line is highlighted by red colour text. See Figure 11.

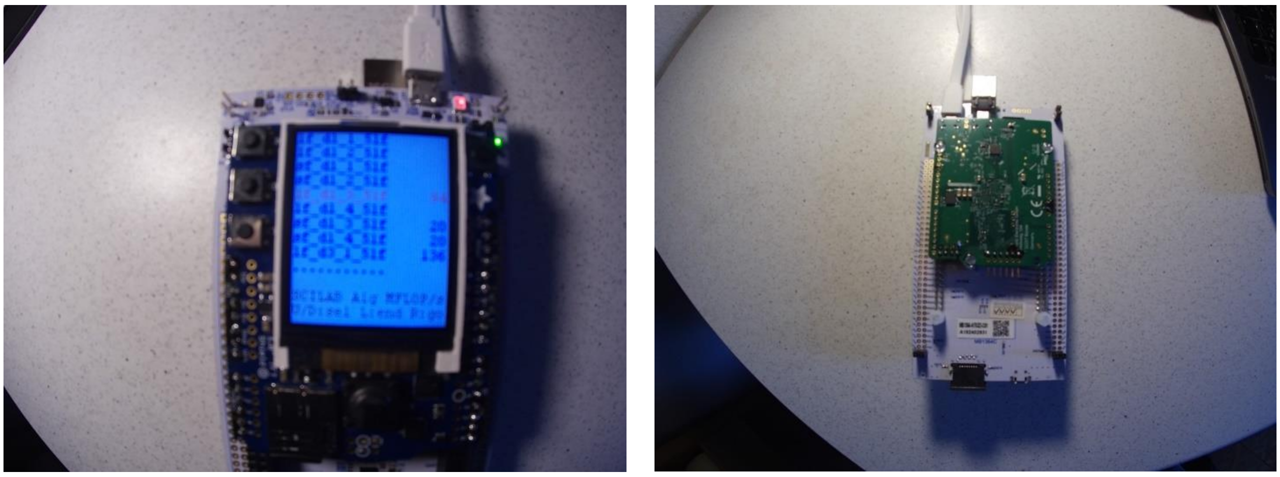

Figure 11: Menu (left) serves for selection of DSP algorithms in Scilab on ArduZynq (right).

2.2. Steps to run menu-driven Scilab demo on terminal:

1. Power ON the ArduZynq by micro USB cable from PC. PetaLinux and Scilab starts.
2. Power ON the NUCLEO-H753ZI board by second micro USB cable from PC.
3. Reset the NUCLEO-H753ZI board by pressing the black button.
4. Bring up the introduction screen by pressing the blue button on the NUCLEO-H753ZI board.
5. Start menu by pressing the joystick DOWN.
6. Scroll up or down in the displayed menu by pressing the joystick DOWN or UP.
7. Execute selected DSP algorithm by pressing the joystick RIGHT.
   Scilab executes the selected DSP algorithm and sends back the measured MFLOP/s information. It is displayed by the terminal on the same line.
8. The terminal function can be terminated by pressing the joystick RIGHT. Scilab is terminated and Linux on ArduZynq is halted. All open files are closed, file system is unmounted from the micro SD card and the power OFF or reset of the ArduZynq shield can be done.
9. Power OFF the NUCLEO-H753ZI board by disconnecting micro USB cable from the PC.
10. Power OFF the ArduZynq shield by disconnecting micro USB cable from the PC.

Warning: Do not power off the micro USB power supply of ArduZynq before execution of step (8). Part of the Linux file system on the micro SD card might be corrupted.

The ArduZynq also supports micro USB FLASH formatted as FAT-32 file system readable by PC. See the 16 GByte USB FLASH in the right part of Figure 11.

Each executed Adaptive system identification benchmark generates data file with results prepared in format readable by PC Matlab. Remove the FLASH disk from the power OFF ArduZynq and display results of each benchmark in graphical form in Matlab a PC.

Figure 12 presents result of adaptive identification with directional forgetting [3]. The time variable system is modelled as a regression filter with time variable parameters.

The top part of Figure 12 presents system input signal with an ill-conditioned section. See the sinus-shape input signal. It is not sufficiently exiting the adaptive identification.

Middle part of Figure 12 presents the time variable regression filter parameters (red) together with actual values delivered as output from the tested, adaptive system identification algorithm with directional forgetting [3] (green).

Bottom part of Figure 12 presents the estimated filtration error delivered as output from the tested, adaptive system identification algorithm with directional forgetting [3] (green).

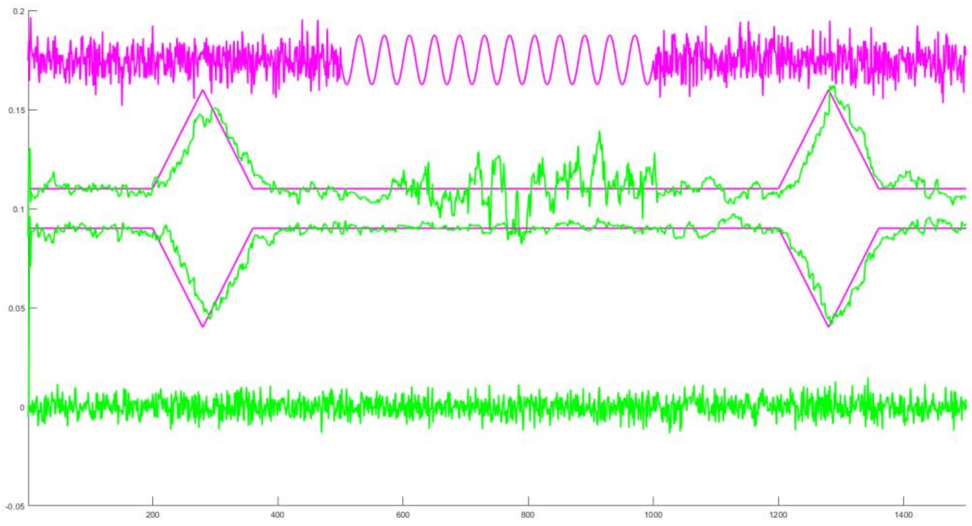

Figure 12: Result of adaptive identification benchmark.

The input signal and the time variable system is modelled in SciLab script in double precision. See Figure 10 and Figure 12.

The generated input/output data are processed by the adaptive recursive identification algorithm [3] written as hand-coded C function with input and output operands defined in Matlab MEX API format.

The adaptive recursive identification algorithm [3] is implemented in double-precision floating point arithmetic or in single-precision floating point arithmetic.

The execution time of the evaluated adaptive recursive identification algorithm is measured after return of results (by value) back into Scilab. The execution time is converted to MFLOP/s by the Scilab script, communicated to the NUCLEO-H753ZI terminal and displayed on the AdaFruit TFT display by the CM7 MCU.

An introduction and overview to adaptive system identification algorithms [1] supported and implemented by UTIA can be found in the application note "Adaptive RLS Algorithms Reference Implementations"

http://sp.utia.cz/results/dsp_1_6/dsp_1_6.pdf

and in the corresponding evaluation package:

http://sp.utia.cz/index.php?ids=results&id=dsp_1_6

2.3. C interface for benchmarks in Scilab

Scilab prepares I/O data and matrices in scripts. Scilab scripts worl with double precision matrices.

Finally, an algorithm written in system identification C MEX function is called from the script. Scilab script calls the system identification function `[...] = md51f(...);`

```
tic;
[u,y,vm,em,efm,udm,ydm,vdm,edm,zum,zym,zvm,zem,pm,dxm,lxm,rm,dfm,fim
,f0m,fam,dzm,zerm,md51_i,md51_d,md51_t,N] = ..
md51f(u,y,vm,em,efm,udm,ydm,vdm,edm,zum,zym,zvm,zem,pm,dxm,lxm,rm,df
m,fim,f0m,fam,dzm,zerm,md51_i,md51_d,md51_t,N);
comp_t = toc();
```

or function `[...] = md51(...);`

```
tic;
[u,y,vm,em,efm,udm,ydm,vdm,edm,zum,zym,zvm,zem,pm,dxm,lxm,rm,dfm,fim
,f0m,fam,dzm,zerm,md51_i,md51_d,md51_t,N] = ..
md51(u,y,vm,em,efm,udm,ydm,vdm,edm,zum,zym,zvm,zem,pm,dxm,lxm,rm,dfm
,fim,f0m,fam,dzm,zerm,md51_i,md51_d,md51_t,N);
comp_t = toc();
```

Scilab is copying input double precision matrix operands to the called C MEX function by value.

MEX function C glue code called by `[...] = md51f(...);` performs conversion of all matrices by value to single precision floating point local arrays and then calls C single precision function `md51f_(...).` Arguments of this C function are are pointers to the local arrays with initial content of matrices defied in single precision floating point:

```
void md51f_(u, y, v, e, ef, ud, yd, vd, ed, zu, zy, zv, ze, p, dx,
     lx, r, df, fi, f0, fa, dzeta, zer, md51_i, md51_d, md51_t, N)
      float *u, *y, *v, *e, *ef, *ud, *yd, *vd, *ed, *zu;
      float *zy, *zv, *ze, *p, *dx, *lx, *r, *df, *fi, *f0;
      float *fa, *dzeta, *zer, *md51_i, *md51_d, *md51_t, *N;
      {...} ;
```

Function `md51f_(...)` works on pre-allocated single-precision matrices “in place” in single precision floating point arithmetic. After return from the function `md51f_(...),` the “glue” Scilab C code converts results by-value back to double-precision floating point. These double-precision results a copied by-value into a new allocated double-precision matrices in Scilab interpret workspace. If matrices with same name and dimensions exist in Scilab, Scilab interpret will clean them and replace them with new allocated result matrices received as copy-by-values from the called MEX interface function `[...] = md51f(...);`.

The interface is simpler in case of the double-precision MEX function. We explain it on call to function `[...] = md51(...);.`

MEX function C glue code called by `[...] = md51(...)`; performs just copy of all matrices by-value to double-precision floating point local arrays and then calls C double-precision function `md51_(...);`.

```
void md51_(u, y, v, e, ef, ud, yd, vd, ed, zu, zy, zv, ze, p, dx,
      lx, r, df, fi, f0, fa, dzeta, zer, md51_i, md51_d, md51_t, N)
      double *u, *y, *v, *e, *ef, *ud, *yd, *vd, *ed, *zu;
      double *zy, *zv, *ze, *p, *dx, *lx, *r, *df, *fi, *f0;
      double *fa, *dzeta, *zer, *md51_i, *md51_d, *md51_t, *N;
      {...} ;
```

Function `md51_(...)` works on pre-allocated, local, double-precision floating point matrices "in place" in double precision arithmetic. After return from the function `md51_(...)`, the "glue" Scilab C code copy double-precision results by-value into new allocated double precision matrices in the Scilab workspace. If matrices with same name and dimensions exist in Scilab, Scilab interpret cleans them and replaces them with new allocated result matrices received as copy-by-values from the C MEX interface function `[...] = md51(...)`;.

In ArduZynq Debian Linux, the C function `md51f_(...)` or the C function `md51_(...)` are linked to Scilab dynamically as shared object libraries `libmex_md51f.so` or `libmex_md51.so`. These libraries have been pre-compiled directly in the embedded C/C++ compiler on the ArduZynq board from C source code.

Scilab interpret script also saves input operands of `md51f_(...)` or `md51_(...)` as floating point or double precision arrays defined in C headers in readable text format. These headers are filled with values used in the Scilab scripts.

This is example of C header `sci_N.h` representing an initial value of Scilab variable `N`

```
static float sci_N[1] = {
1.500000000000e+03
};
```

2.4. C interface for benchmarks on STM32H755ZI-Q

C Projects with benchmarks for CM7 and CM4 MCUs on STM32H755ZI-Q board use C header data created in Scilab and call pre-compiled static library `libmd51d_CM7.a`, `libmd51f_CM7.a` or `libmd51f_CM4.a` . Libraries contain compiled double precision function `md51_()` for CM7 MCU or single precision function `md51f_()` for CM7 or CM4 target MCUs.

The C code used for compiation of `md51_()` and `md51f_()` function is identical in case of Scilab shared object libraries on ArduZynq and in the case of static libraries for benchmark projects executed by CM7 or CM4 MCUs on the STM32H755ZI-Q board.

2.5. Performance on ArduZynq and STM32H755ZI-Q

This section presents measured MFLOP/s results for C coded DSP algorithms implemented:

- SciLab MEX C function running on ArduZynq shield in the NUCLEO-H753 Terminal.
- Project for STM32H755ZI-Q board CM7 or CM4 MCU.

Performance results in MFLOP/s for the adaptive RLS QRD Inverse-update identification of time variable stochastic SISO system are listed for CM7 MCU in Figure 13, Figure 14 and for CM4 MCU in Figure 15.

top_51d.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascale+ ZU03-CG Double precision MFLOP/s 1200 MHz	Ardu Zynq XC7Z010 Double precision MFLOP/s 650 MHz	H755ZI-Q **CM7 MCU** Double precision MFLOP/s 400 MHz	**Adaptive RLS QRD Inverse-update** identification of time variable stochastic SISO system. 1500 steps. Without square root. EF=Exponential forgetting; DF=Directional forgetting [1], [3], [6]
lf_d1_1_51d	159	81	**36.4**	FIR filter, order: 23, EF
lf_d1_2_51d	167	86	**37.8**	FIR filter, order: 23, DF
sf_d1_1_51d	48	20	**18.0**	FIR filter, order: 3, EF
sf_d1_2_51d	48	20	**18.0**	FIR filter, order: 3, DF
lf_d1_3_51d	159	81	**36.4**	FIR filter, order: 23, EF, ill cond. I/O
lf_d1_4_51d	159	86	**37.8**	FIR filter, order: 23, DF, ill cond. I/O
sf_d1_3_51d	48	20	**18.0**	FIR filter, order: 3, EF, ill cond. I/O
sf_d1_4_51d	48	20	**18.0**	FIR filter, order: 3, DF, ill cond. I/O
lf_d3_1_51d	211	112	**39.8**	FIR filter, order: 83, EF

Figure 13: DP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM7

top_51f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascale+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM7** Single Precision MFLOP/s 400 MHz	**Adaptive RLS QRD Inverse-update** identification of time variable stochastic SISO system. 1500 steps. Without square root. EF=Exponential forgetting; DF=Directional forgetting [1], [3], [6]
lf_d1_1_51f	189	97	**68.7**	FIR filter, order: 23, EF
lf_d1_2_51f	189	100	**70.3**	FIR filter, order: 23, DF
sf_d1_1_51f	48	20	**28.8**	FIR filter, order: 3, EF
sf_d1_2_51f	48	20	**28.8**	FIR filter, order: 3, DF
lf_d1_3_51f	189	97	**68.7**	FIR filter, order: 23, EF, ill cond. I/O
lf_d1_4_51f	177	97	**70.3**	FIR filter, order: 23, DF, ill cond. I/O
sf_d1_3_51f	48	20	**28.8**	FIR filter, order: 3, EF, ill cond. I/O
sf_d1_4_51f	48	20	**28.8**	FIR filter, order: 3, DF, ill cond. I/O
lf_d3_1_51f	277	136	**86.1**	FIR filter, order: 83, EF

Figure 14: SP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM7

Performance of the 400 MHz CM7 MCU in floating point is close to the performance of the 650 MHz A9 MPU. The ArduZynq A9 Scilab RLS QRD inverse-update identification algorithms are negatively affected systems with small number of parameters. It is due to the communication overhead related to copy-by-value of parameters from Scilab to C and back.

In case of system with 83 parameters the single precision performance corresponds to relation of clocks of both compared systems (1200 MHz / 650 MHz / 400 MHz).

top_51f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM4** Single Precision MFLOP/s 200 MHz	**Adaptive RLS QRD Inverse-update** identification of time variable stochastic SISO system. 100 steps on CM4. Without square root. EF=Exponential forgetting; DF=Directional forgetting [1], [3], [6]
lf_d1_1_51f	189	97	**11.2**	FIR filter, order: 23, EF
lf_d1_2_51f	189	100	**11.8**	FIR filter, order: 23, DF
sf_d1_1_51f	48	20	**4.8**	FIR filter, order: 3, EF
sf_d1_2_51f	48	20	**4.8**	FIR filter, order: 3, DF
lf_d1_3_51f	189	94	**11.2**	FIR filter, order: 23, EF, ill cond. I/O
lf_d1_4_51f	177	97	**11.8**	FIR filter, order: 23, DF, ill cond. I/O
sf_d1_3_51f	48	20	**4.8**	FIR filter, order: 3, EF, ill cond. I/O
sf_d1_4_51f	48	20	**4.8**	FIR filter, order: 3, DF, ill cond. I/O
lf_d3_1_51f	277	136	**13.5**	FIR filter, order: 83, EF

Figure 15: SP FP MFLOP/s: RLS QRD Inverse-update on ArduZynq and CM4

Performance results in MFLOP/s for the adaptive RLS QRD identification of time variable stochastic SISO system are listed for CM7 MCU in Figure 16, Figure 17 and Figure 18.

top_76d.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Double precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Double precision MFLOP/s 650 MHz	H755ZI-Q **CM7** Double Precision MFLOP/s 400 MHz	**Adaptive RLS QRD** identification of time variable stochastic SISO system. 1500 steps. Algorithm without square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_76d	146	82	**36,0**	FIR filter, order: 23, EF
sf_d1_1_76d	39	19	**17.3**	FIR filter, order: 3, EF
lf_d1_3_76d	146	81	**36.0**	FIR filter, order: 23, EF, ill cond. I/O
sf_d1_3_76d	39	20	**18.0**	FIR filter, order: 3, EF, ill cond. I/O
lf_d3_1_76d	170	93	**37.4**	FIR filter, order: 83, EF

Figure 16: DP FP MFLOP/s: RLS QRD identification on ArduZynq and CM7

top_76f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM7** Single Precision MFLOP/s 400 MHz	**Adaptive RLS QRD** identification of time variable stochastic SISO system. 1500 steps. Algorithm without square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_76f	159	90	**72.0**	FIR filter, order: 23, EF
sf_d1_1_76f	52	22	**26.0**	FIR filter, order: 3, EF
lf_d1_3_76f	159	93	**72.0**	FIR filter, order: 23, EF, ill cond. I/O
sf_d1_3_76f	52	22	**26.0**	FIR filter, order: 3, EF, ill cond. I/O

lf_d3_1_76f	185	116	**80.7**	FIR filter, order: 83, EF

Figure 17: SP FP MFLOP/s: RLS QRD identification on ArduZynq and CM7

In case of adaptive RLS QRD identification without square root are results similar to the inverse update.

The performance of the 200 MHz CM4 MCU in floating point for RLS QRD inverse update and RLS QRD is taking advantage of the single precision floating point HW unit. See Figure 15 and Figure 18. The performance of the 200 MHz CM4 MCU in floating point for RLS QR with square root functions has to implement this function in SW and this explains why the performance of CM4 MCU is lower. See Figure 21.

top_76f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM4** Single Precision MFLOP/s 200 MHz	**Adaptive RLS QRD** identification of time variable stochastic SISO system. 1500 steps. Algorithm without square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_76f	159	90	**10.2**	FIR filter, order: 23, EF
sf_d1_1_76f	52	22	**3.5**	FIR filter, order: 3, EF
lf_d1_3_76f	159	93	**10.2**	FIR filter, order: 23, EF, ill cond. I/O
sf_d1_3_76f	52	22	**3.5**	FIR filter, order: 3, EF, ill cond. I/O

Figure 18: SP FP MFLOP/s: RLS QRD identification on ArduZynq and CM4

Performance results in MFLOP/s for the adaptive RLS QRD identification of time variable stochastic SISO system are listed for CM7 MCU in Figure 19, Figure 20 and Figure 21.

top_77d.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Double precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Double precision MFLOP/s 650 MHz	H755ZI-Q **CM7** Double Precision MFLOP/s 400 MHz	**Adaptive RLS QR** identification of time variable stochastic SISO system. 1500 steps. Algorithm with square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_77d	163	89	**39,3**	FIR filter, order: 23, EF
sf_d1_1_77d	48	24	**19.2**	FIR filter, order: 3, EF
lf_d1_3_77d	158	89	**39.3**	FIR filter, order: 23, EF, ill cond. I/O
sf_d1_3_77d	64	24	**19.2**	FIR filter, order: 3, EF, ill cond. I/O
lf_d3_1_77d	191	101	**41.6**	FIR filter, order: 83, EF

Figure 19: DP FP MFLOP/s: RLS QR identificationon ArduZynq and CM7

top_77f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM7** Single Precision MFLOP/s 400 MHz	**Adaptive RLS QR** identification of time variable stochastic SISO system. 1500 steps. Algorithm with square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_77f	176	96	**80.5**	FIR filter, order: 23, EF
sf_d1_1_77f	64	24	**27.4**	FIR filter, order: 3, EF
lf_d1_3_77f	175	99	**80.5**	FIR filter, order: 23, EF, ill cond. I/O

sf_d1_3_77f	48	27	**27.4**	FIR filter, order: 3, EF, ill cond. I/O
lf_d3_1_77f	209	118	**94.3**	FIR filter, order: 83, EF

Figure 20: SP FP MFLOP/s: RLS QR identification on ArduZynq and CM7

top_77f.sce Adapt sys. & C headers generated by scripts:	Zynq Ultrascal+ ZU03-CG Single precision MFLOP/s 1200 MH	Ardu Zynq XC7Z010 Single precision MFLOP/s 650 MHz	H755ZI-Q **CM4** Single Precision MFLOP/s 200 MHz	**Adaptive RLS QR** identification of time variable stochastic SISO system. 1500 steps. Algorithm with square root function calls. EF=Exponential forgetting; [1], [3], [6]
lf_d1_1_77f	176	96	**6.1**	FIR filter, order: 23, EF
sf_d1_1_77f	64	24	**1.7**	FIR filter, order: 3, EF
lf_d1_3_77f	175	99	**6.1**	FIR filter, order: 23, EF, ill cond. I/O
sf_d1_3_77f	48	27	**1.7**	FIR filter, order: 3, EF, ill cond. I/O

Figure 21: SP FP MFLOP/s: RLS QR identification on ArduZynq and CM4

2.6. Projects released for STM32H755ZI-Q board

Projects for STM32H755ZI-Q board CM7 and CM4 MCU are using the STM32Cube_FW_H7_V1.5.0 development framework and the AC6 System Workbench for STM32. Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

• **libmd51d_CM7.a**	DP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd51f_CM7.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd51f_CM4.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd76d_CM7.a**	DP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd76f_CM7.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd76f_CM4.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd77d_CM7.a**	DP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd77f_CM7.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3
• **libmd77f_CM4.a**	SP FP STM32H755ZITx; Board: NUCLEO-H755ZI-Q; -O3

Simulation I/O data for the STM32H755ZI-Q CM7 and CM4 MCU projects have been generated in Scilab as set of C header files specific for each target project.
See Figure 22, Figure 23 and Figure 24. See supported boards and displays in Figure 25.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-1-CM7

STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-4-CM7
Single Precision Floating Point - CM4 MCU - Inverse QRD Identification [1], [3], [6]; 100 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-2-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-3-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d1-4-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-lf-d3-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-2-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-3-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51f-sf-d1-4-CM4

Figure 22: Inverse QRD identification projects, STM32H755ZI-Q, CM7, CM4 MCUs

Double Precision Floating Point - CM7 MCU QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76d-sf-d1-3-CM7
Single Precision Floating Point - CM7 MCU - QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-sf-d1-3-CM7
Single Precision Floating Point - CM4 MCU - QRD Identification [1], [3], [6]; 100 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-lf-d1-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-lf-d1-3-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-sf-d1-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-76f-sf-d1-3-CM4

Figure 23: QRD identification projects (without sqrt), STM32H755ZI-Q, CM7, CM4 MCUs

Double Precision Floating Point - CM7 MCU QR Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77d-sf-d1-3-CM7
Single Precision Floating Point - CM7 MCU - QR Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-sf-d1-3-CM7
Single Precision Floating Point - CM4 MCU - QR Identification [1], [3], [6]; 100 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-lf-d1-1-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-lf-d1-3-CM4
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-sf-d1-1-CM4

STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-77f-sf-d1-3-CM4

Figure 24: QR identification projects (with sqrt) STM32H755ZI-Q, CM7, CM4 MCUs

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-H755ZIQ MB1363C Adafruit V1	USE_STM32H7XX_NUCLEO_144_MB1363 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=1
NUCLEO-H755ZIQ **MB1363C** **Adafruit V2**	**USE_STM32H7XX_NUCLEO_144_MB1363** **ADAFRUIT_TFT_JOY_SD_ID802** **TFTSHIELD_VERSION=2**

Figure 25: Combinations of boards and displays supported by released projects

Projects are released for the 32/64 bit System Workbench for STM32 AC6 and the STM32Cube_FW_H7_V1.5.0 framework.

2.7. Performance of STM32 F767, H7A3, H743/53, H723

Performance results in MFLOP/s for the adaptive RLS QRD Inverse-update identification benchmarks defined in Figure 13, Figure 14 is summarised for A53 MPU, A9 MPU and for STM32F767ZI, STM32H7AZI-Q, STMH743ZI, STM32H753ZI and STM32H723ZG MCUs in Figure 26 and Figure 27.

top_51d.sce Adapt sys. & C headers generated by scripts:	Zynq Us+ **A53** 1200 MHz Double precision MFLOP/s	ArduZynq **A9** 650 MHz Double precision MFLOP/s	F767 CM7 216 MHz Double precision MFLOP/s	H7A3 CM7 280 MHz Double precision MFLOP/s	H743/53 CM7 400 MHz Double precision MFLOP/s	H723 CM7 520 MHz Double precision MFLOP/s
lf_d1_1_51d	159	81	**19.5**	**25.2**	**36,4**	**46,5**
lf_d1_2_51d	167	86	**20.3**	**26.3**	**37.8**	**48.8**
sf_d1_1_51d	48	20	**9.6**	**12.0**	**18.0**	**20,6**
sf_d1_2_51d	48	20	**9.6**	**13.1**	**18.0**	**24,0**
lf_d1_3_51d	159	81	**19.5**	**36.4**	**36.4**	**46,5**
lf_d1_4_51d	159	86	**20.3**	**37.8**	**37.8**	**48,8**
sf_d1_3_51d	48	20	**9.6**	**12.0**	**18.0**	**20,6**
sf_d1_4_51d	48	20	**9.6**	**13.1**	**18.0**	**24,0**
lf_d3_1_51d	211	112	**22.6**	**28.9**	**39.8**	**55,3**

Figure 26: SP MFLOP/s: Comparison of A53, A9 MPUs with F767, H7A3, H743. H753, H723 MCUs.

top_51f.sce Adapt sys. & C headers generated by scripts:	Zynq Us+ **A53** 1200 MHz Single precision MFLOP/s	ArduZynq **A9** 650 MHz Single precision MFLOP/s	F767 CM7 216 MHz Single precision MFLOP/s	H7A3 CM7 280 MHz Single precision MFLOP/s	H743/53 CM7 400 MHz Single precision MFLOP/s	H723 CM7 520 MHz Single precision MFLOP/s
lf_d1_1_51f	189	97	**36.9**	**47.3**	**68.7**	**88,9**
lf_d1_2_51f	189	100	**38.3**	**49.6**	**70.3**	**91,6**
sf_d1_1_51f	48	20	**14.4**	**18.0**	**28.8**	**28,8**
sf_d1_2_51f	48	20	**14.4**	**18.0**	**28.8**	**36,0**
lf_d1_3_51f	189	97	**36.9**	**47.3**	**68.7**	**88,9**
lf_d1_4_51f	177	97	**38.3**	**49.6**	**70.3**	**91,6**
sf_d1_3_51f	48	20	**14.4**	**18.0**	**28.8**	**28,8**
sf_d1_4_51f	48	20	**14.4**	**18.0**	**28.8**	**36,0**
lf_d3_1_51f	277	136	**47.7**	**60.9**	**86.1**	**114,7**

Figure 27: DP MFLOP/s: Comparison of A53, A9 MPUs with F767, H7A3, H743, H743, H753, H723 MCUs.

Next sections describe released projects for the STMH743ZI, STM32H753ZI, STM32F767ZI, STM32H7AZI-Q and STM32H732ZG MCUs.

2.8. Released projects for STM32H743

Projects for STM32H743ZI board with CM7 MCU are using the STM32Cube_FW_H7_V1.5.0 development framework and the AC6 System Workbench for STM32. Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

• **libmd51d_CM7.a**	DP FP STM32H743ZITx; Board: NUCLEO-H743ZI2; -O3
• **libmd51f_CM7.a**	SP FP STM32H743ZITx; Board: NUCLEO-H743ZI2; -O3

See Figure 28. See supported board and display combinations in Figure 29.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H743ZI2\Benchmark-test-51f-sf-d1-4-CM7

Figure 28: Inverse QRD identification projects, STM32H743ZI2, CM7 MCU 400 MHz

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-H743ZI2 MB1364 Adafruit V1	USE_STM32H7XX_NUCLEO_144_MB1364 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=1
NUCLEO-H743ZI2 MB1364 Adafruit V2	**USE_STM32H7XX_NUCLEO_144_MB1364 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=2**
NUCLEO-H743ZI MB1137 Rev B Adafruit V1	USE_STM32H7XX_NUCLEO_144 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=1
NUCLEO-H743ZI MB1137 Rev B Adafruit V2	USE_STM32H7XX_NUCLEO_144 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=2

Figure 29: Combinations of boards and displays supported by released projects

Figure 29 describes supported combinations of STM evaluation boards and displays. Released projects can be configured by change of compiler defines. Bold defines are the default values present in the default configuration of released projects. Four combinations are supported for the STM32H743ZI MCU.

Projects are released for the 32/64 bit System Workbench for STM32 AC6 and the STM32Cube_FW_H7_V1.5.0 framework.

2.9. Released projects for STM32H753

Released projects for STM32H753ZI board with CM7 MCU are using the STM32Cube_FW_H7_V1.5.0 development framework and the AC6 System Workbench for STM32. Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

- **libmd51d_CM7.a** DP FP STM32H753ZITx; Board: NUCLEO-H753ZI; -O3
- **libmd51f_CM7.a** SP FP STM32H753ZITx; Board: NUCLEO-H753ZI; -O3

See Figure 30. For supported board and display see Figure 25.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H753ZI\Benchmark-test-51f-sf-d1-4-CM7

Figure 30: Inverse QRD identification projects, STM32H753ZI, CM7 MCU 400 MHz

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-H753ZI MB1364 Adafruit V1	USE_STM32H7XX_NUCLEO_144_MB1364 ADAFRUIT_TFT_JOY_SD_ID802 TFTSHIELD_VERSION=1
NUCLEO-H753ZI **MB1364** **Adafruit V2**	**USE_STM32H7XX_NUCLEO_144_MB1364** **ADAFRUIT_TFT_JOY_SD_ID802** **TFTSHIELD_VERSION=2**

Figure 31: Combinations of boards and displays supported by released projects

Projects are released for AC6 System Workbench for STM32 and the STM32Cube_FW_H7_V1.5.0 framework.

2.10. Released projects for STM32F767

Released projects for STM32F767ZI board with CM7 MCU are using the STM32Cube_FW_F7_V1.15.0 development framework and the AC6 System Workbench for STM32. Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

• **libmd51d_CM7.a**	DP FP STM32F767ZITx; Board: NUCLEO-F767ZI; -O3
• **libmd51f_CM7.a**	SP FP STM32F767ZITx; Board: NUCLEO-F767ZI; -O3

See Figure 32. See supported board and display combinations in Figure 33.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-lf-d1-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_F7_V1.15.0\Projects\STM32F767ZI-Nucleo\Benchmark-test-51f-sf-d1-4-CM7

Figure 32: Inverse QRD identification projects, STM32F767ZI, CM7 MCU 216 MHz

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-F767ZI	**USE_STM32H7XX_NUCLEO_144**
MB1137B	**ADAFRUIT_TFT_JOY_SD_ID802**
Adafruit V1	**TFTSHIELD_VERSION=1**

Figure 33: Combination of board and display supported by released projects

Projects are released for the 32/64 bit System Workbench for STM32 AC6 and the STM32Cube_FW_F7_V1.15.0 framework.

2.11. Released projects for STM32H7A3

Projects for STM32H7A3ZI-Q board with CM7 MCU are using the STM32Cube_FW_H7_V1.9.0 development framework and the STM32CubeIDE 1.7.0 integrated development environment.

Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

• **libmd51d_CM7.a**	DP FP STM32H7A3ZITx-Q; Board: NUCLEO-F7A3ZI-Q; -O3
• **libmd51f_CM7.a**	SP FP STM32H7A3ZITx-Q; Board: NUCLEO-F7A3ZI-Q; -O3

See Figure 34. See supported board and display combinations in Figure 35.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H7A3ZI-Q\Benchmark-test-51f-sf-d1-4-CM7

Figure 34: Inverse QRD identification projects, STM32H7A3ZI-Q, CM7 MCU 280 MHz

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-H7A3ZI-Q **MB1363D** **Adafruit V1**	

Figure 35: Combination of board and display supported by released projects

Projects for the STM32H7A3ZI-Q MCU have been released for the STM32CubeIDE version 1.7.0 and the STM32Cube_FW_H7_V1.9.0 framework. The STM32Cube_FW_H7_V1.9.0 projects are using component drivers supporting only the Adafruit_Shield V1.

2.12. Released projects for STM32H723ZG

Projects for STM32H723ZG board with CM7 MCU are using the STM32Cube_FW_H7_V1.9.0 development framework and the STM32CubeIDE 1.7.0 integrated development environment.

Projects are linked with the Adaptive RLS QRD Inverse-update identification algorithm provided by UTIA in form of precompiled static libraries:

• **libmd51d_CM7.a**	DP FP STM32H23ZG; Board: NUCLEO-F723ZG; -O3
• **libmd51f_CM7.a**	SP FP STM32H23ZG; Board: NUCLEO-F723ZG; -O3

See Figure 34. See supported board and display combinations in Figure 35.

Double Precision Floating Point - CM7 MCU Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-lf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-lf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-lf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-lf-d1-4-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-lf-d3-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-sf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-sf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-sf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51d-sf-d1-4-CM7
Single Precision Floating Point - CM7 MCU - Inverse QRD Identification [1], [3], [6]; 1500 steps
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-lf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-lf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-lf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-lf-d1-4-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-lf-d3-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-sf-d1-1-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-sf-d1-2-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-sf-d1-3-CM7
STM32Cube_FW_H7_V1.9.0\Projects\NUCLEO-H723ZG\Benchmark-test-51f-sf-d1-4-CM7

Figure 36: Inverse QRD identification projects, STM32H7A3ZI-Q, CM7 MCU 280 MHz

PCB Board + Display	Define for compiler to be set by user **(BOLD is default)**
NUCLEO-H723ZG **MB1364E** **Adafruit V1**	

Figure 37: Combination of board and display supported by released projects

Projects for the STM32H723ZG MCU have been released for the STM32CubeIDE version 1.7.0 and the STM32Cube_FW_H7_V1.9.0 framework. The STM32Cube_FW_H7_V1.9.0 projects are using component drivers supporting only the Adafruit_Shield V1.

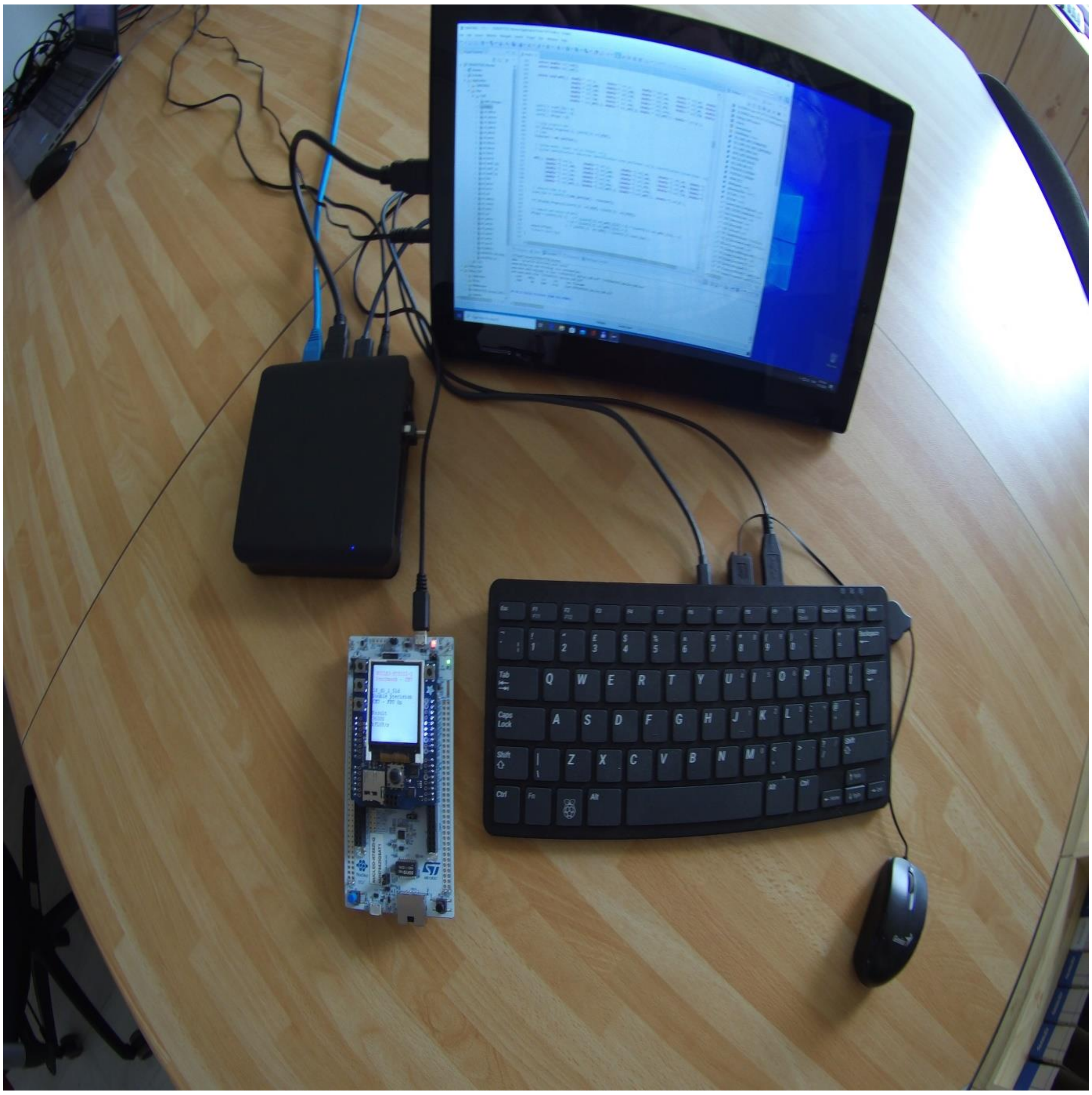
Figure 38: Double precision system identification, STM32H755ZI-Q board, CM7 MCU

Figure 38 presents test of the double-precision system identification project: STM32Cube_FW_H7_V1.5.0\Projects\NUCLEO-H755ZI-Q\Benchmark-test-51d-lf-d1-1-CM7

Project was compiled in the STM32Cube_FW_H7_V1.5.0 on PC with AC6 system workbench for STM32. Compiled code is running on standard, unmodified, NUCLEO-STM32H755ZI-Q evaluation board with the Adafruit V2 display. The "clasic" Adafruit V1 display is also supported by the project. Measured double precision floating point performance of the CM7 MCU is printed on the TFT display.

3 CONCLUSION

This application note describes terminal for menu-driven selection of Scilab benchmarks. The terminal is using the NUCLEO-H753ZI board with ArduZynq shield and AdaFruit TFT shield for the GUI.

Benchmarks perform adaptive RLS QRD Inverse-update identification [3] of time variable stochastic SISO system in single-precision and double-precision floating point.

Scilab interpret was used to generate I/O data environment for execution of these benchmarks on STM32H755ZI-Q board with CM7 and CM4 MCUs.

Performance of

- ArduZynq A9 MPU (650 MHz)
- Zynq Ultrascale+ A53 MPU (1200 MHz)

is compared with

- STM32H755ZI-Q CM7 MCU (400 MHz)
- STM32H755ZI-Q CM4 MCU (200 MHz)
- STM32F767ZI CM7 MCU (216 MHz),
- STM32H7AZI-Q CM7 MCU (280 MHz),
- STM32H743ZI CM7 MCU (400 MHz),
- STM32H753ZI CM7 MCU (400 MHz)
- STM32H723ZG CM7 MCU (520 MHz)

Performance of the 400 MHz CM7 MCU is higher in comparison to the 200 MHz CM4 MCU due to the presence of the fpv5-d16 unit on CM7 MCU supporting both the SP and DP floating point. The fpv4-d16 unit present on CM4 MCU supports only the SP floating point.

Benchmarks indicate that the CM7 MCU (400 MHz) can perform adaptive RLS QRD Inverse-update identification [3] of time variable stochastic SISO system with these maximal sampling rates:

Adaptive Inverse QRD System Identification	**Sampling time**	**Sampling frequency**
Order 83, Double precision FP	0.557 ms	1.79 kHz
Order 83, Single precision FP	0.256 ms	3.90 kHz
Order 23, Double precision FP	0.056 ms	17.85 kHz
Order 23, Single precision FP	0.029 ms	34.12 kHz

The final application target for the presented adaptive inverse QRD System Identification algorithm will be the self-tuning controller [1]-[2].

The CM7 MCU on the STM32H755ZI-Q board will perform the adaptive controller [1]-[2] based on the recursive invers QRD system identification [3].

I/O data acquisition will be performed by the CM4 MCU with the on-chip ADC and DAC converters of the STM32H755ZI device.

Performed benchmarks indicate the potentially achievable sampling rates for the recursive identification part [1], [3] of an adaptive controller [2].

REFERENCES

[1] Peterka, Václav: Bayesian approach to system identification. In Trends and Progress in System identification, IFAC Series for Graduates, Research Workers, Pergamon Press 1981.

[2] Peterka, Václav: Control of uncertain processes. Applied theory and algorithms. In.: Kybernetika (22), Academia [1986].

[3] Kulhavý Rudolf : Directional Tracking of Regression-Type Model Parameters. In Preprints of the 2-nd IFAC Workshop on Adaptive Systems in Control and Signal processing. Lund. S. 97-102. [1986].

[4] Nedoma Petr, Kadlec Jiří, Schier Jan: Tools for Implementation of Parallel Algorithms for Adaptive Control and Signal Processing , 4th IFAC International Symposium on Adaptive Systems in Control and Signal Processing. ACASP '92, p. 727-730 , Eds: Landau I. D., Dugard L., M'Saad M., Laboratoire d'Automatique, (Grenoble 1992) , IFAC International Symposium on Adaptive Systems in Control and Signal Processing. ACASP '92 /4./, (Grenoble, FR, 01.07.1992-03.07.1992) [1992]

[5] Kadlec Jiří: A Joint Criterion for Exponential Directional and Mixed Parameter Tracking, 4th IFAC International Symposium on Adaptive Systems in Control and Signal Processing. ACASP '92, p. 687-692 , Eds: Landau I. D., Dugard L., M'Saad M., Laboratoire d'Automatique, (Grenoble 1992) , IFAC International Symposium on Adaptive Systems in Control and Signal Processing. ACASP '92 /4./, (Grenoble, FR, 01.07.1992-03.07.1992) [1992]

[6] Kulhavý Rudolf, Zarrop M. B.: On a General Concept of Forgetting, International Journal of Control vol.58, 4 (1993), p. 905-924 [1993]

DISCLAIMER