FRONTGRADE

APPLICATION NOTE

UT32M0R500-EVB

Users Guide

5/20/2021
Version #: 1.1.0

Product Name	Manufacturer Part Number	SMD #	Device Type
Arm Cortex M0+	UT32M0R500	5962-17212	01

Table 1: Cross Reference of Applicable Products

1.0 Introduction

The UT32M0R500-EVB Development Board provides a comprehensive and rapid prototyping platform for the UT32M0R500 Microcontroller. The Arduino™ Uno connectivity and full product pinout allow for easy expansion and accessibility. Along with the microcontroller, the subject board supports an external clock, includes JTAG connectors for debugging, and USB-to-UART connectors for communicating from a PC.

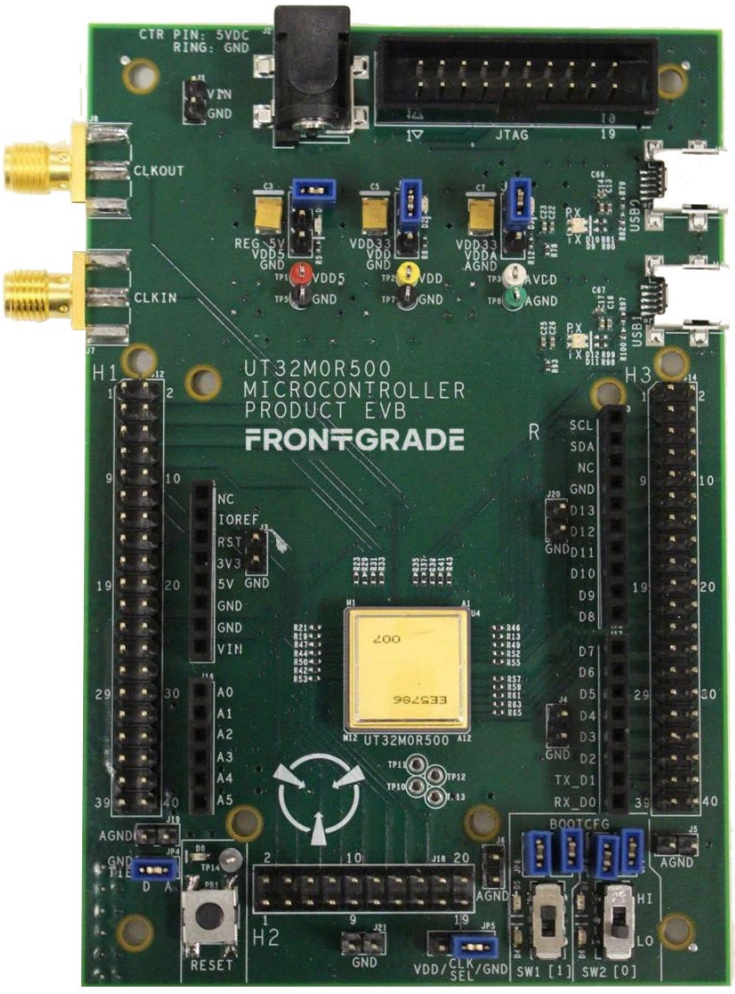


Figure 1: UT32M0R500 Evaluation Board

2.0 Reference Documents

Description	Reference Document
UT32M0R500 Datasheet	https://frontgrade.com/sites/default/files/documents/Datasheet-UT32M0R500.pdf
ARM Keil ULINK2 Hardware Debugger	http://www2.keil.com/mdk5/ulink

3.0 Block Diagram Description and Picture

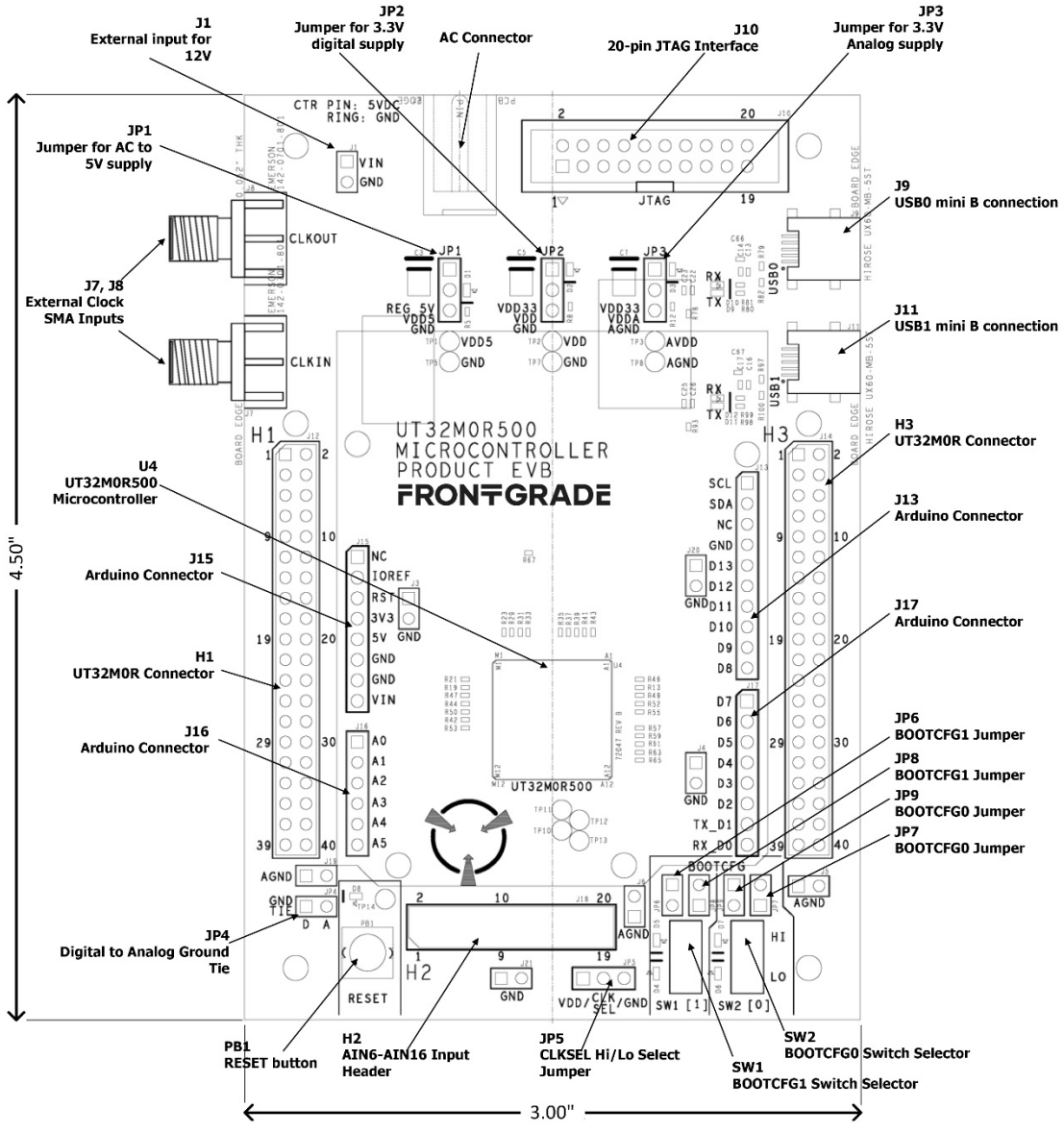


Figure 2: UT32M0R500 EVB Description

4.0 Jumper and Switch Setting Summary

Jumper	Setting	Description/Comments
JP1	Shunt Pin 1 to 2 → Connect 3.3V Digital Supply	Shunt to provide power to board from the VIN pin. If implemented, make sure to disconnect the AC wall plug.
JP2	Shunt Pin 1 to 2 → Connect 3.3V Digital Supply	Required for device operation
JP3	Shunt Pin 1 to 2 → Connect 3.3V Analog Supply	Required for device operation
JP4	Shunt Pin D to A	Connects digital and analog grounds
JP5	Shunt Pin 1 to 2 → Connect CLKSEL pin to VDD Shunt Pin 2 to 3 → Connect CLKSEL pin to GND	Shunt required for proper operation, see Clock Source Options
JP6	Shunt for BOOTCFG1 → Connect LEDs to VDD	Provides power to LEDs connected to BOOTCFG1 and power for SW1. JP8 also required.
JP7	Shunt for BOOTCFG0 → Connect SW0 to U4	JP9 also required.
JP8	Shunt for BOOTCFG1 → Connect SW1 to U4	JP6 also required.
JP9	Shunt for BOOTCFG0 → Connect LEDs to VDD	Provides power to LEDs connected to BOOTCFG0 and power for SW1. JP7 also required.
J1	External connection for 12V input to VIN signal	
J7	SMA connector for CLKOUT signal	JP5 must be connected to VDD when applying clock signal to this pin.
J8	SMA connector for CLKIN signal	JP5 must be connected to VDD when applying clock signal to this pin.
J9	USB mini-B connector for USB0	Connection for communicating to U4 over USB through UART0
J10	20-pin JTAG interface connector	
J11	USB mini-B connector for USB1	Connection for communicating to U4 over USB through UART1
J13	Arduino connector	
J14	Arduino connector	
J16	Arduino connector	
J17	Arduino connector	
H1(J12)	Connector for U4	
H2(J18)	20-pin header for AIN6 – AIN15 input	
H3(J14)	Connector for U4	
SW1	Toggle switch for BOOTCFG1 selection	
SW2	Toggle switch for BOOTCFG0 selection	
PB1	RESET Button	
U4	UT32M0R500 device	

5.0 Power Supply and Power Selections

The UT32M0R500-EVB has two options for providing power to the board. The first option is to provide power via the AC wall supply provided with the development board. The second option is to provide 7V – 12V to VIN (J1).

NOTE: Make sure you only use one of the mentioned methods

6.0 Boot Configuration Options

The UT32M0R500 device has three different boot modes configured through the BOOTCFG pins. The UT32M0R500EVB supports the control of the BOOTCFG pins through two methods. The first method is via SW1 and SW0. To use SW1 and SW0, jumpers JP6 – JP9 must be in place. The second method that allows for control of the BOOTCFG pins is through H2 (pins 17 and 19).

Boot Mode Selection Pins		Boot Mode	Description
BOOTCFG1	BOOTCFG2		
0	0	0	Load image from internal Flash memory into SRAM and execute
0	1	1	Reserved
1	0	2	Load/Update image over UART0 into flash (reset required)
1	1	3	Load/Update image over CAN0 into flash (reset required)

NOTE: For control through H2, make sure to remove jumper JP6 – JP9.

7.0 Clock Source Options

The UT32M0R500-EVB supports all clocking options for the UT32M0R500 microcontroller. There is the option to use the internal clock source or use an external source. This is determined by the CLKSEL pin, which is controlled by JP5. For the external clock source, the UT32M0R500 can utilize a clock signal (square wave with 50% duty cycle) or crystal oscillator input. In the case of the external clock source, the UT32M0R500-EVB can support a clock source by connecting to the SMA connectors (J7 and J8). Another option is populate the board with a crystal oscillator and support circuitry.

CLKSEL	Description
0	Selects internal clock
1	Selects the External Crystal Source External clock of crystal oscillator or clock signal on CLKIN support

8.0 Programming and Debugging Interface

The UT32M0R500-EVB supports programming and communicating with the microcontroller over UART. For programming the microcontroller, the UART0 peripheral is used. To facilitate communicating over UART from a PC, the EVB includes to USB-to-UART converters connected on USB0 and USB1 for UART0 and UART1 respectively. Where both UARTs can be utilized for communication, only UART0 (via USB0) can be used for programming.

The UT32M0R500-EVB supports debugging through the 20-pin JTAG (J10) interface. To program the UT32M0R500 over JTAG only the ARMKeil ULINK2 hardware debugger is officially supported.

8.1 Creating a Project with Keil µVision IDE

1. Launch Keil uVision
2. From the Project menu, select New uVision Project….
3. Under the directory of choice, specify the project name as **helloword** and click Save, see Figure 3.

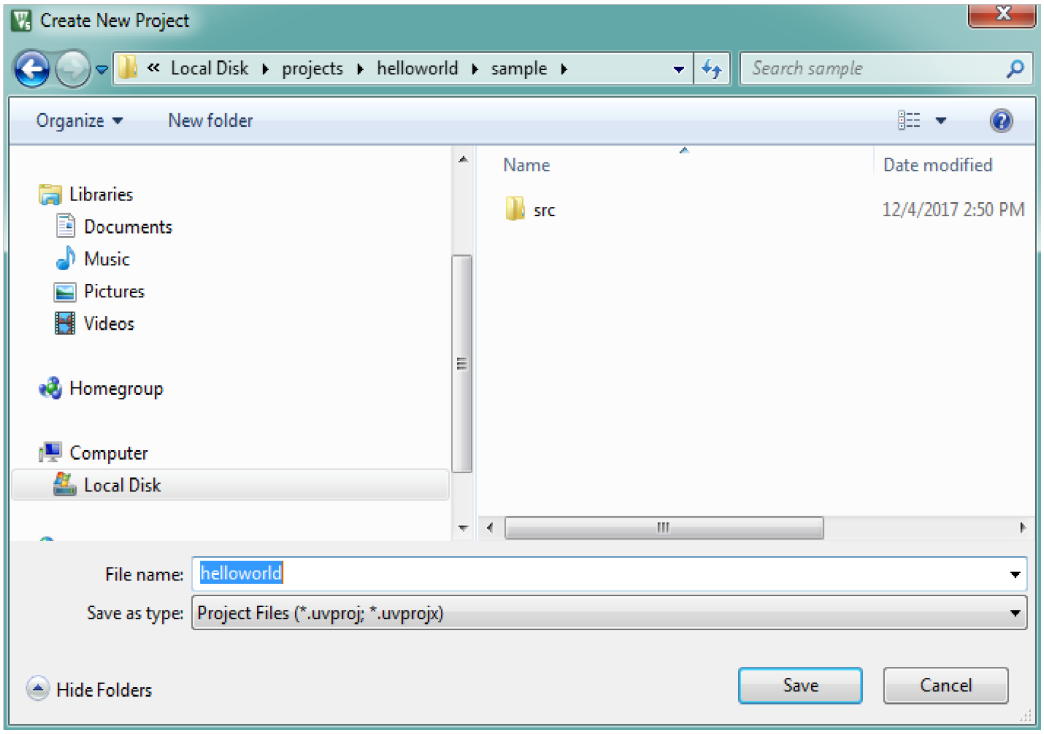


Figure 3: Project Setup

4. Select **Device** and click **OK,** see figure 4.

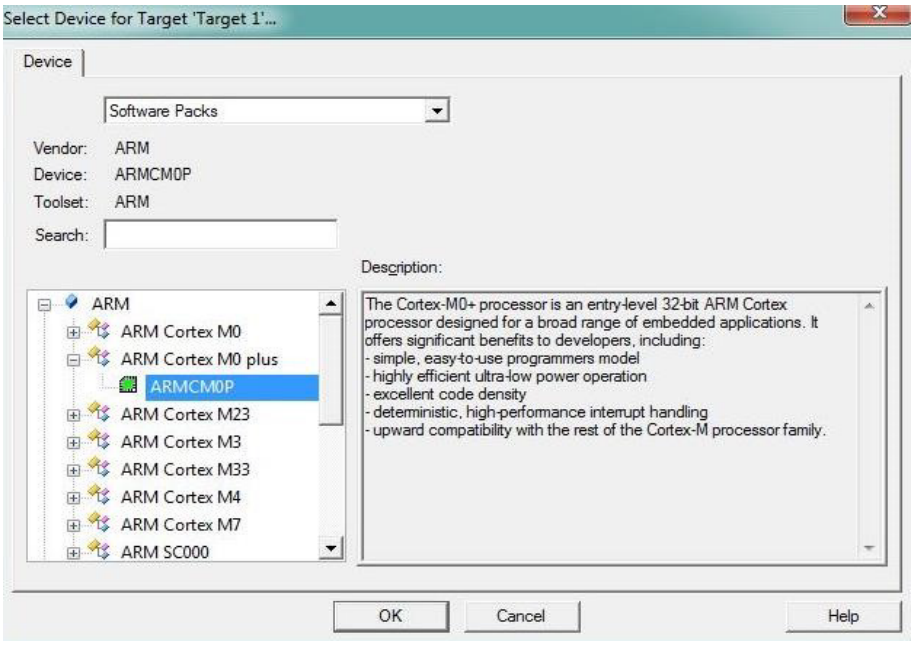


Figure 4: Select Device

5. Click the **Manage Run-Time Environment** symbol  and under **Software Component**, select the appropriate components and click **OK**, see Figure 5.

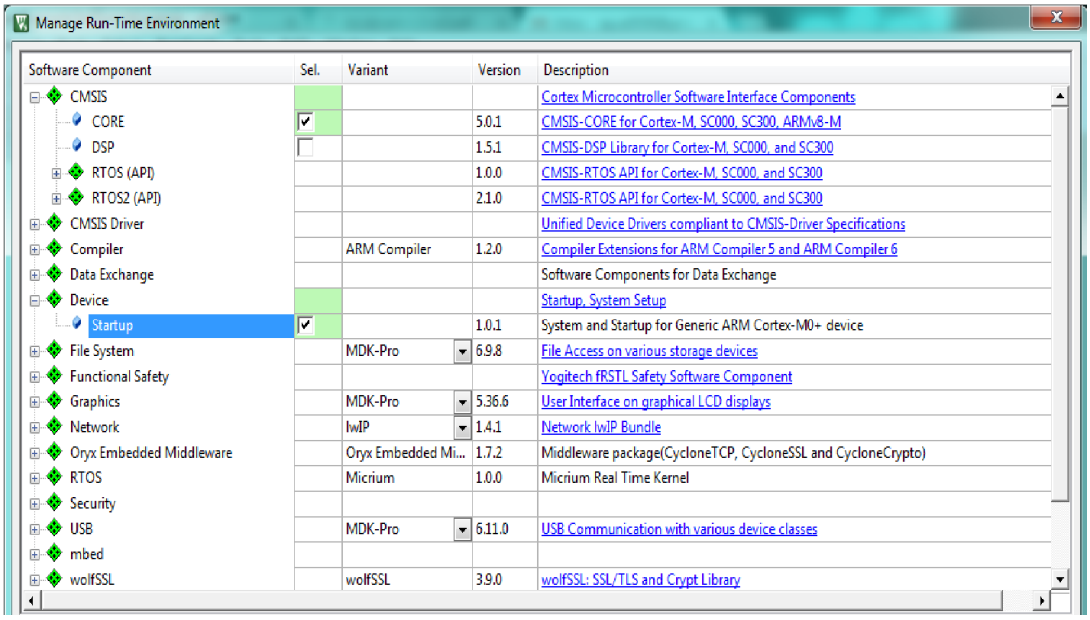


Figure 5: Software Components

6. Under the folder where the project was created, create a **src** folder for the **.c** files. In the **Project**, double-click **Source Group 1** and rename it to **hello_src**.
7. Right-click on **hello_src** and click on **Add New Item** to **Group 'hello_src'**…. Add a new **C** source file, **hello_test.c** and copy the following code.

```
#include <stdio.h>
#include "UT32M0R500.h"
#include "ut32m0_uart.h"

UART_TypeDef *UART0 = (UART_TypeDef *) UART0_BASE;
UART_InitTypeDef UART_InitStruct;
uint32_t ActualBaudRate;

int main (void){
    UART_StructInit (&UART_InitStruct);
    ActualBaudRate=UART_Init (UART0, &UART_InitStruct);
    UART_Cmd (UART0, ENABLE, ENABLE);
          for(;;){
          printf("Hello World!!!\r\n");
          }
}
```

8. Right-click on **Target1** and select **Add Group…** to create groups for source and include files for Frontgrade's Standard Peripheral Library. Add sources and include files to their respective directories, see **Figure 6**

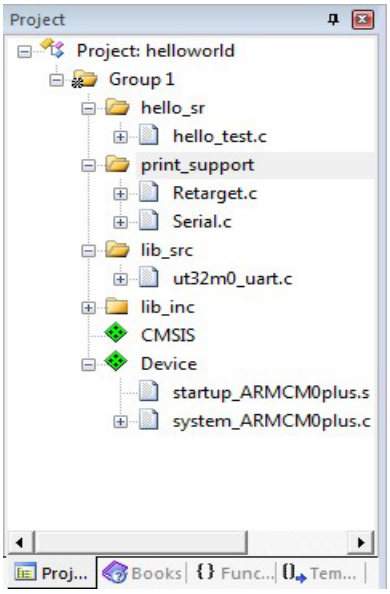


Figure 6: Add Source and Include Files

9. Right-click on **Target1** and select **Options for Target 'Target 1'….** see **Figures 7-17** for basic settings—

   Change setting according to the particular project. For **C/C++** and **Asm** tabs, click ... and setup the compiler include paths; see **Figure 11** and **Figure 12.** Leave the other tabs with defaults.

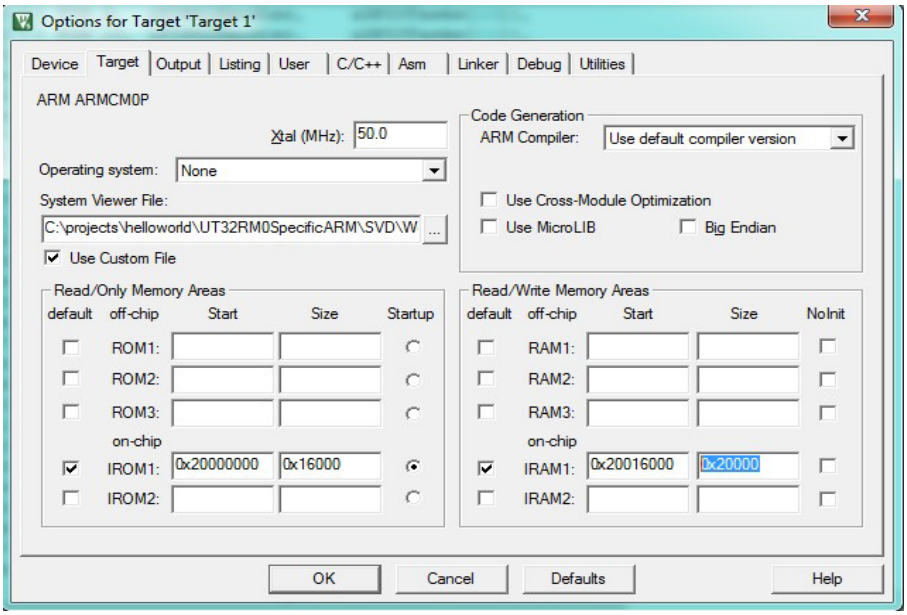


Figure 7: Target

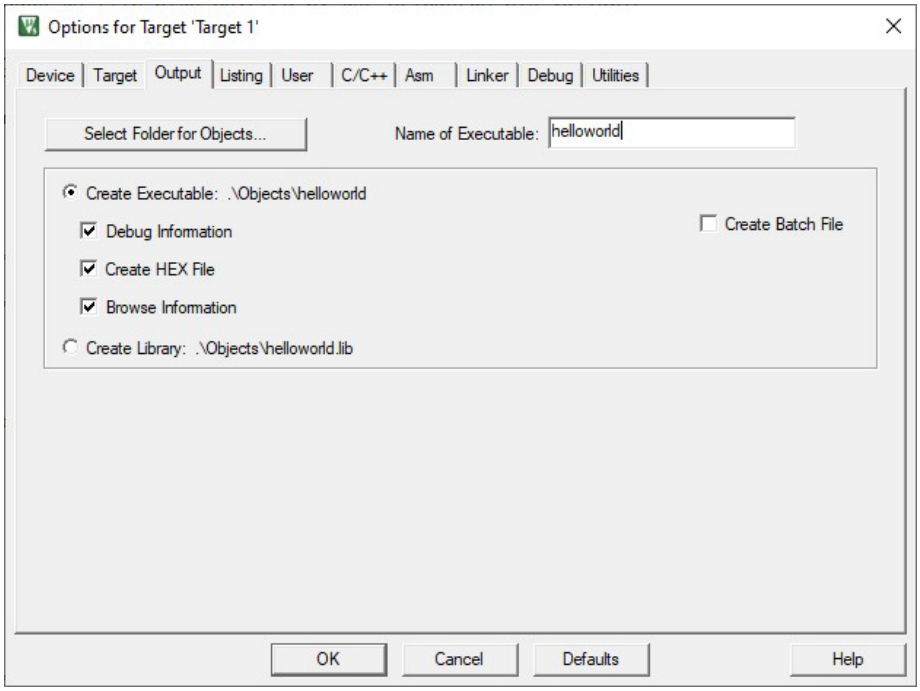


Figure 8: Output

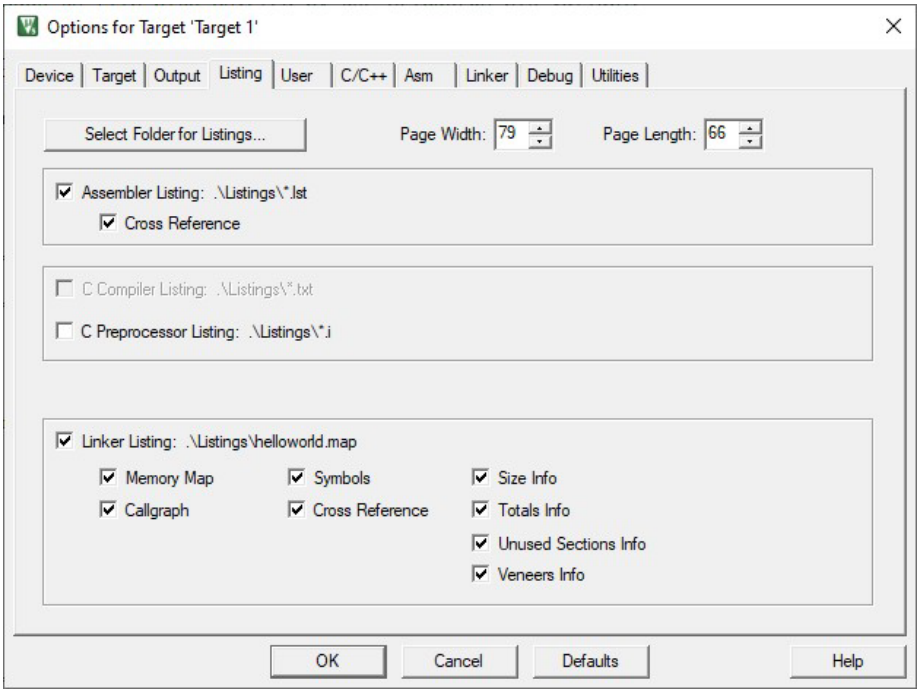


Figure 9: Listing

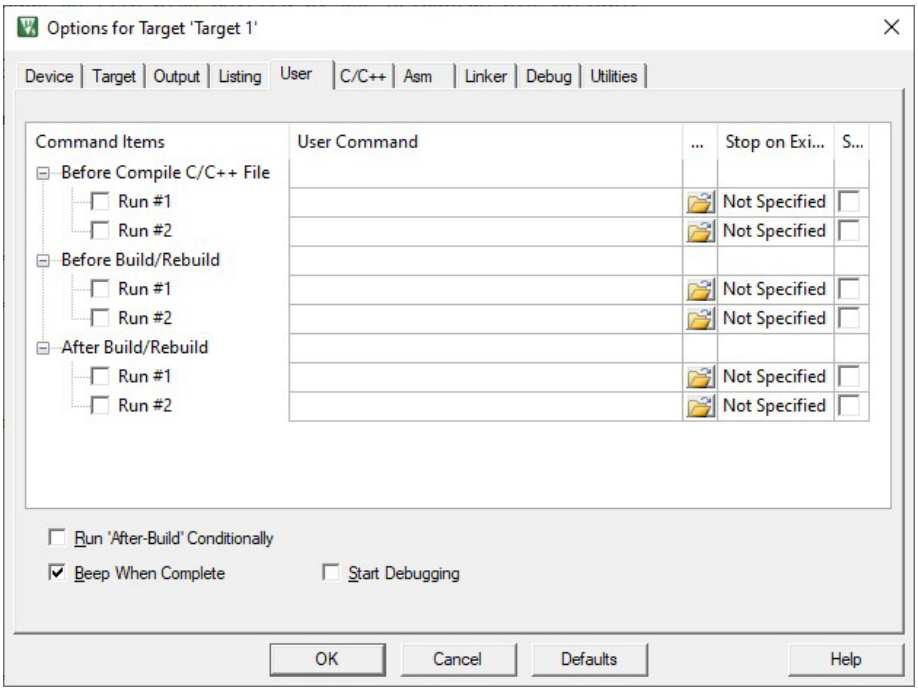


Figure 10: User

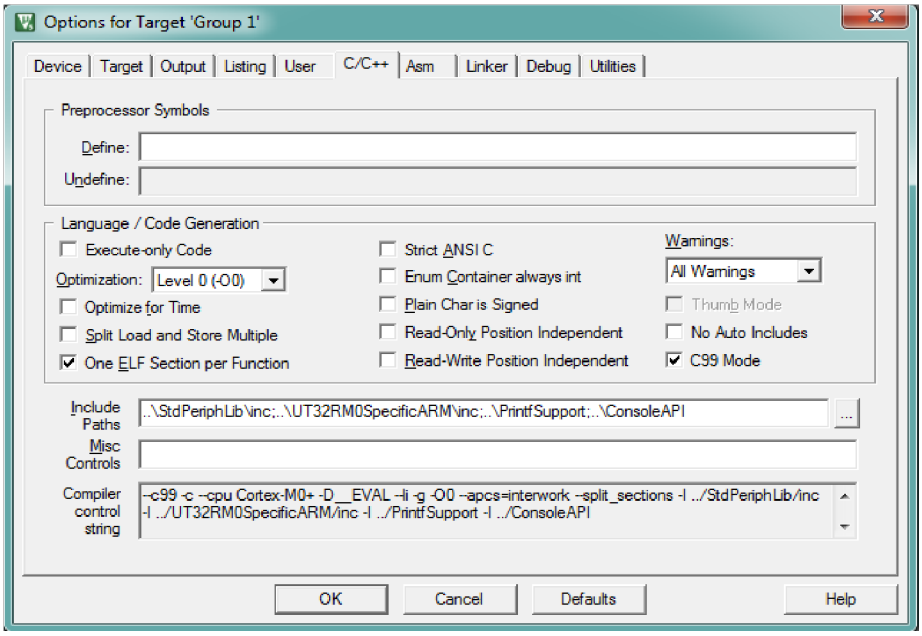


Figure 11: C/C++ Include Paths

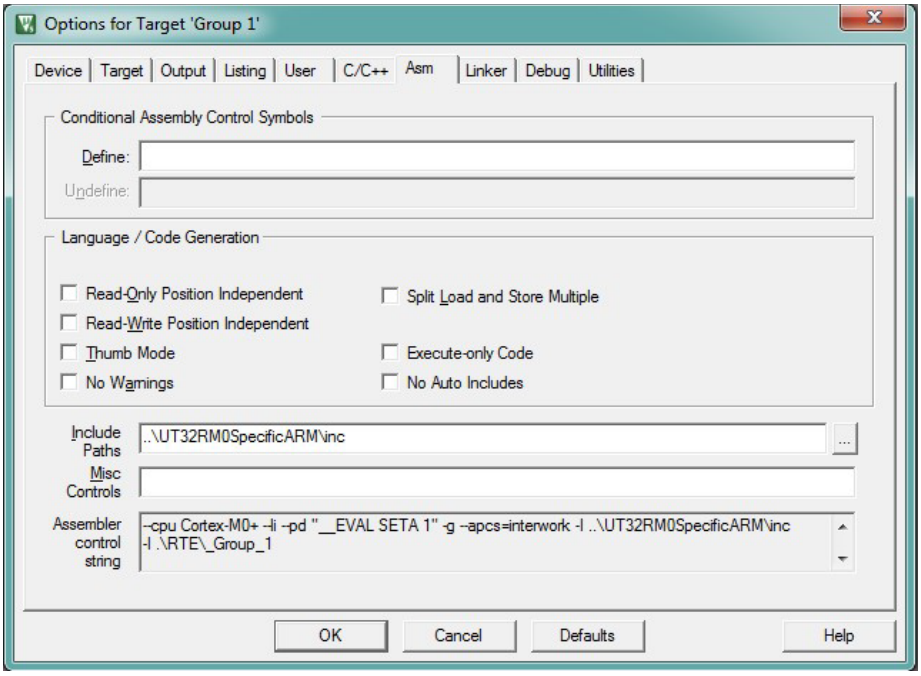


Figure 12: ASM Include Paths

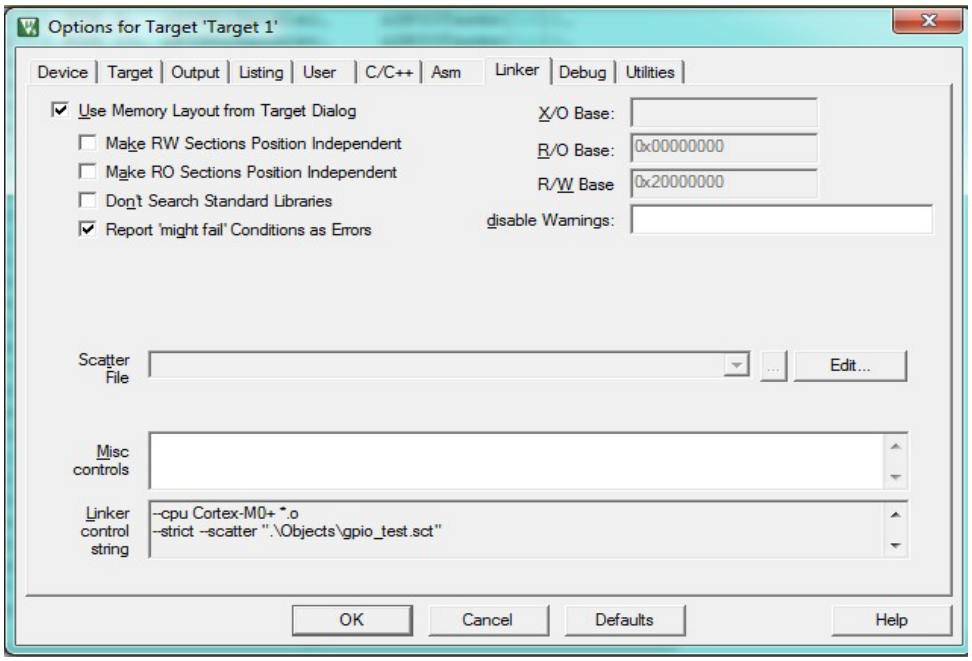


Figure 13: Linker

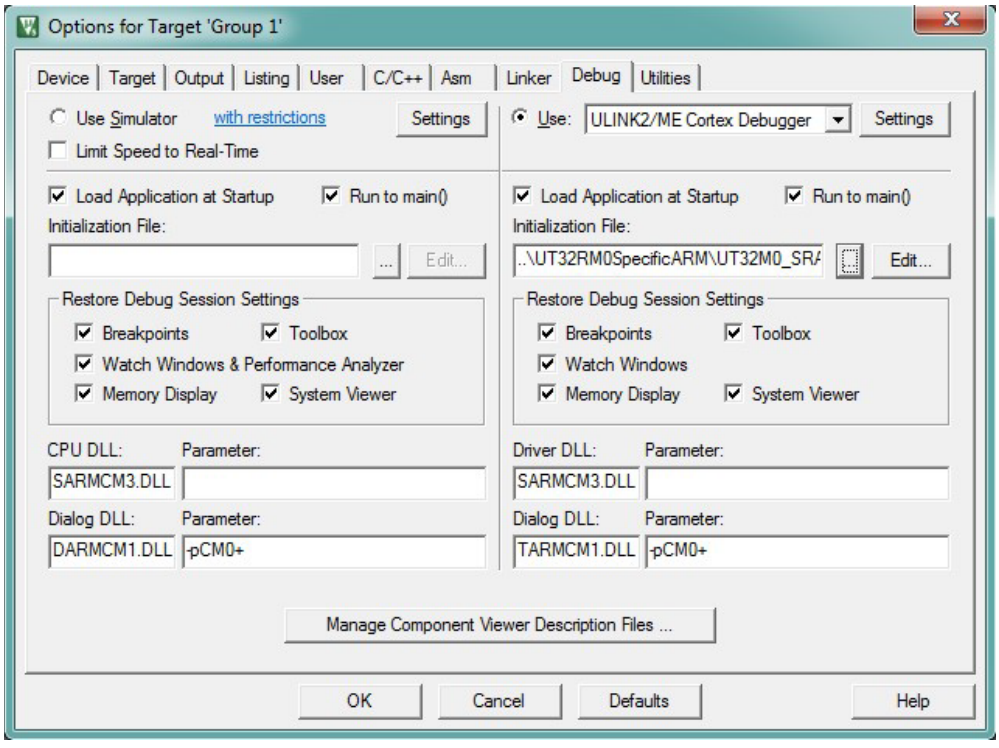


Figure 14: Debug

To get to **Figures 15-17,** click **"Settings"** in the top right of **Figure 14**

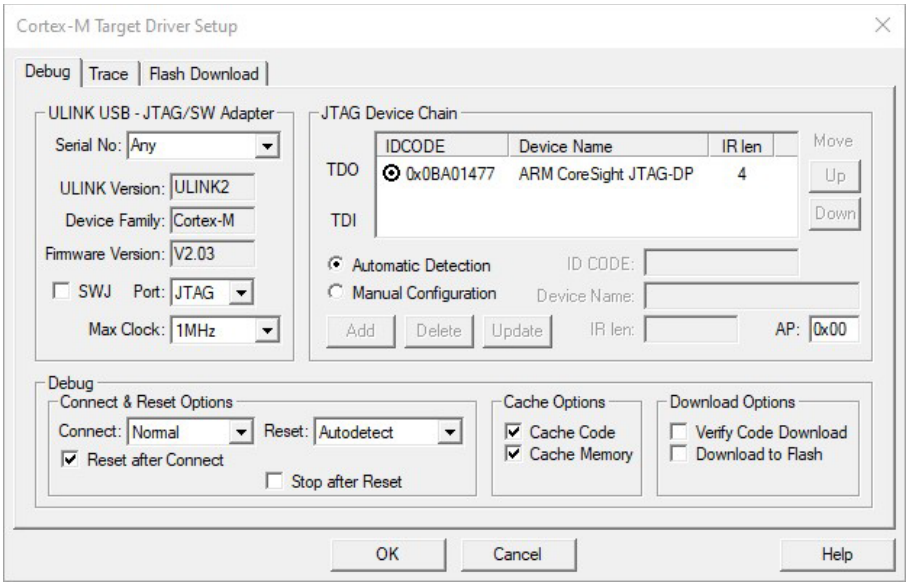


Figure 15: Debug JTAG Settings

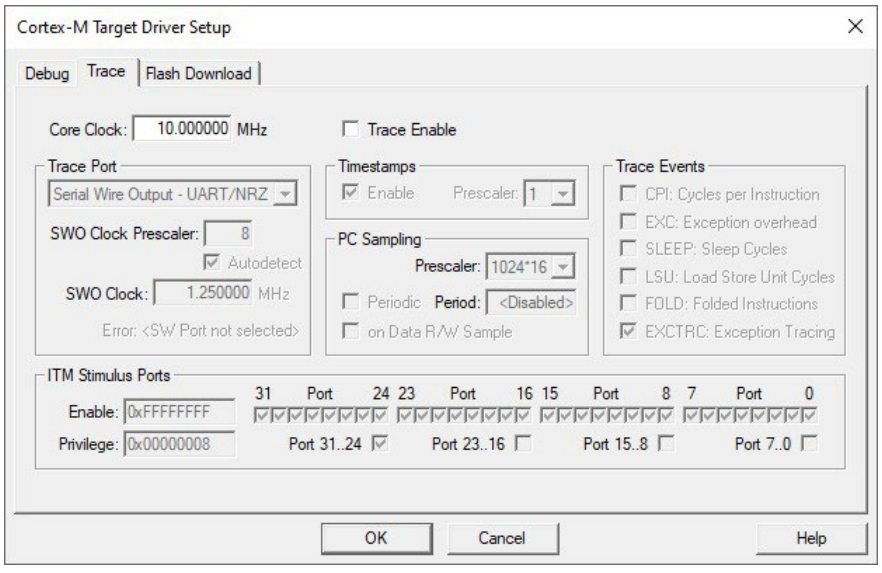


Figure 16: Trace JTAG Settings

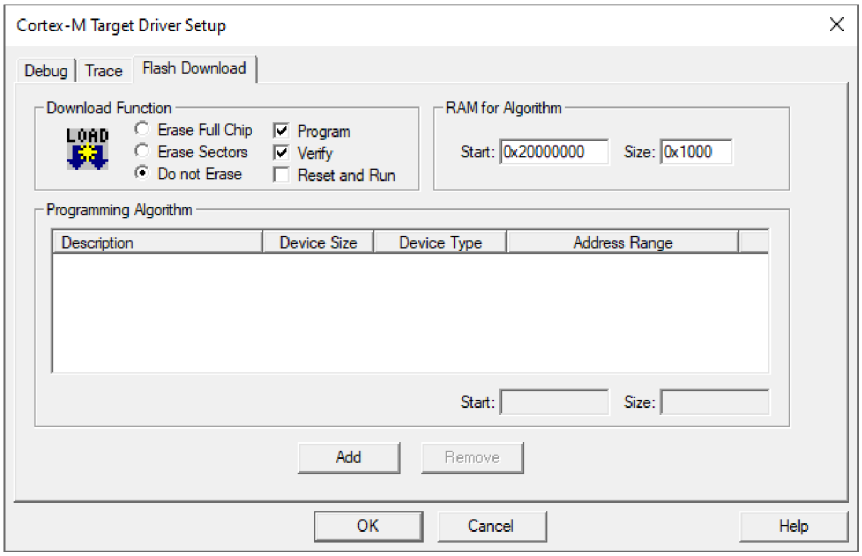


Figure 17: Flash Download JTAG Settings

10. In the Project Explorer view, click on and **Build Project.**

11. Start the debugger and run the application. Display the output using a Terminal, see **Figure 12.**

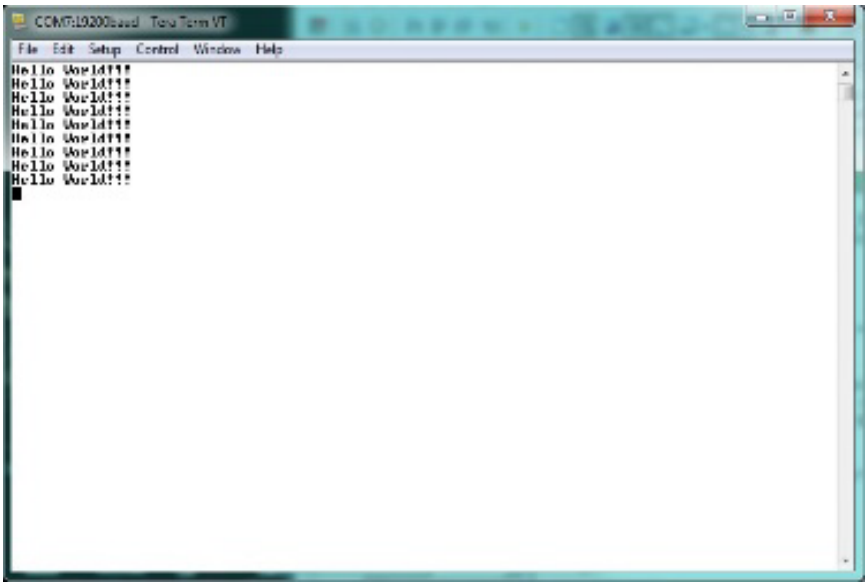


Figure 18: Hello World Display

8.2 UART Terminal Configuration

To program the UT32M0R500 via UART, confirm that the switches for BOOTCFG are in the b'10 position. This enables the UART interface (UART0: UART0_RXD and UART0_TXD) peripheral. The UART0 is initialized to operate at 19200 baud, with 8 databits, 1 stop bit, no parity, and flow control off.

During the UART0 firmware update process, the UT32M0R500's BootRom expects an Intel Hex record file to be uploaded. Depending on the host terminal emulator, the carriage return (0x0D) and line feed (0x0A) characters may be deleted in each line. These characters are required for successful upload. To avoid this, it is recommended that the Intel Hex record be uploaded in the 'binary' mode (as opposed to ASCII).

To prevent overrunning the UT32M0R500's UART receiver, two features need to be enabled prior to file upload. First, "line pacing" should be set to 10 milliseconds (ms). Second, XON/XOFF (software) flow control should be enabled during the update.

8.3 Uploading a HEX File Via UART

Before sending an Intel Hex record file, ensure you have a proper connection established by pushing the RESET button (PB1). You should see the following output:

```
Updating from UART...

Welcome to the Cobham AES UT32M0R500 BootROM
Enter a '?' for the user menu

:>
```

You can now see the menu of commands by sending '?' or just hitting return. For this example, the Intel Hex record file will be written into the NOR Flash in image 0. To access the NOR Flash, send 'DEV -tN' to set your target device. Then, send 'IMG -n0' to select image 0.

```
:>IMG -n0

NFC init SUCCESS!
```

To ensure previous uploads don't interfere with this upload, send the command 'ERS' to erase the current image. To check if 'ERS' was successful, send the command 'VFY'. Don't worry about any ERROR message, just make sure that Embedded = 0xFFFF. You can now program your board with 'PGM –fH'. You will see:

```
:>PGM -fH

Send/upload image (hex) file now.
Be sure to enable all THREE of the following features:
  (a) XON/OFF software flow control
  (b) 10ms line pacing
  (c) binary mode
```

You can upload your file (check the above features are enabled). If the file uploads successfully, you will see:

```
:>.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
Programming complete -- check progress stream for any 'E/1/2/3' (errors)
```

If the upload has no errors, send VFY again. Take note of the Calculated value.

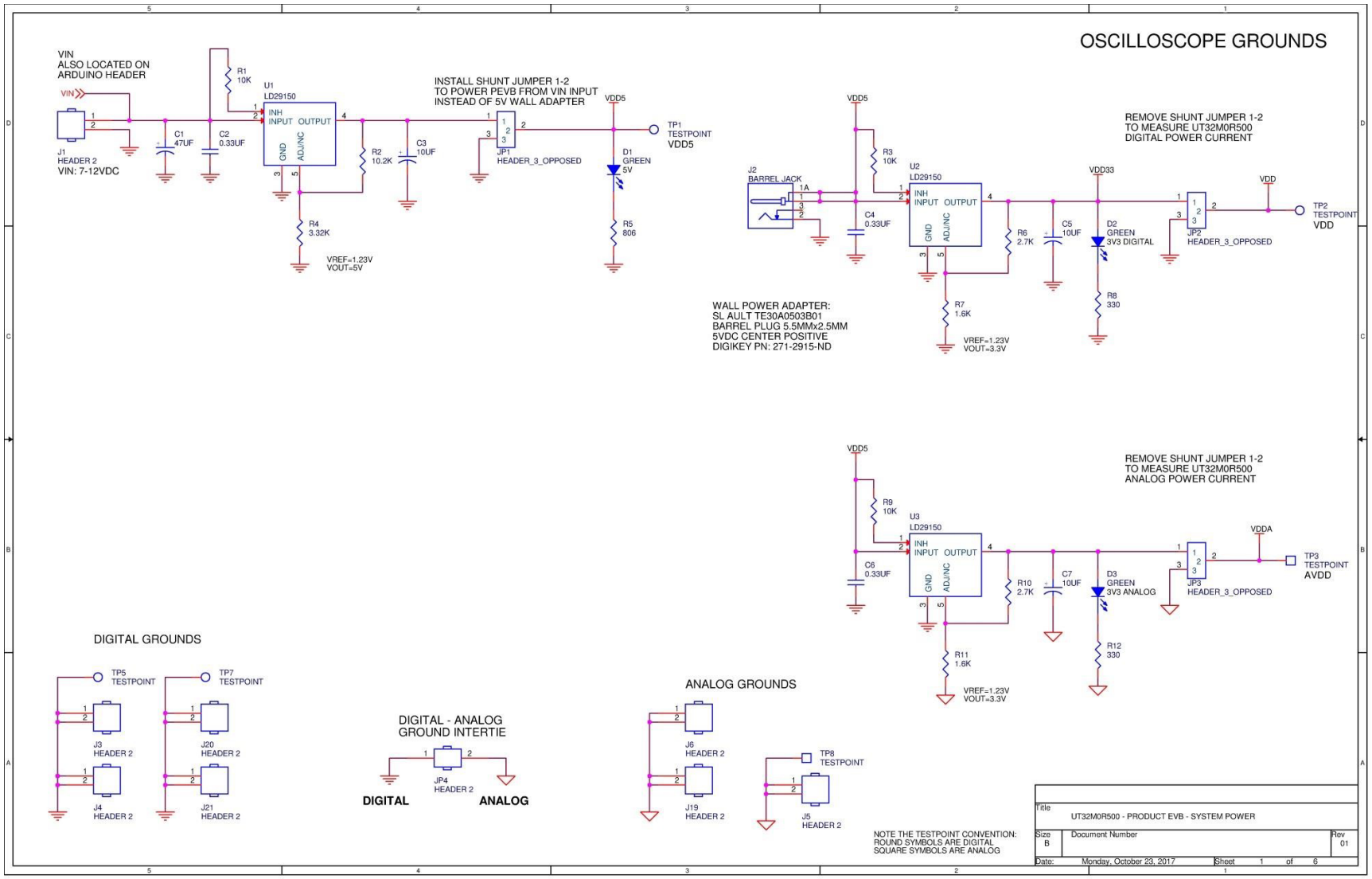
```
:>VFY
ERROR: CRC mismatch. Calculated = 0xCDF1, Embedded = 0xFFFF
```

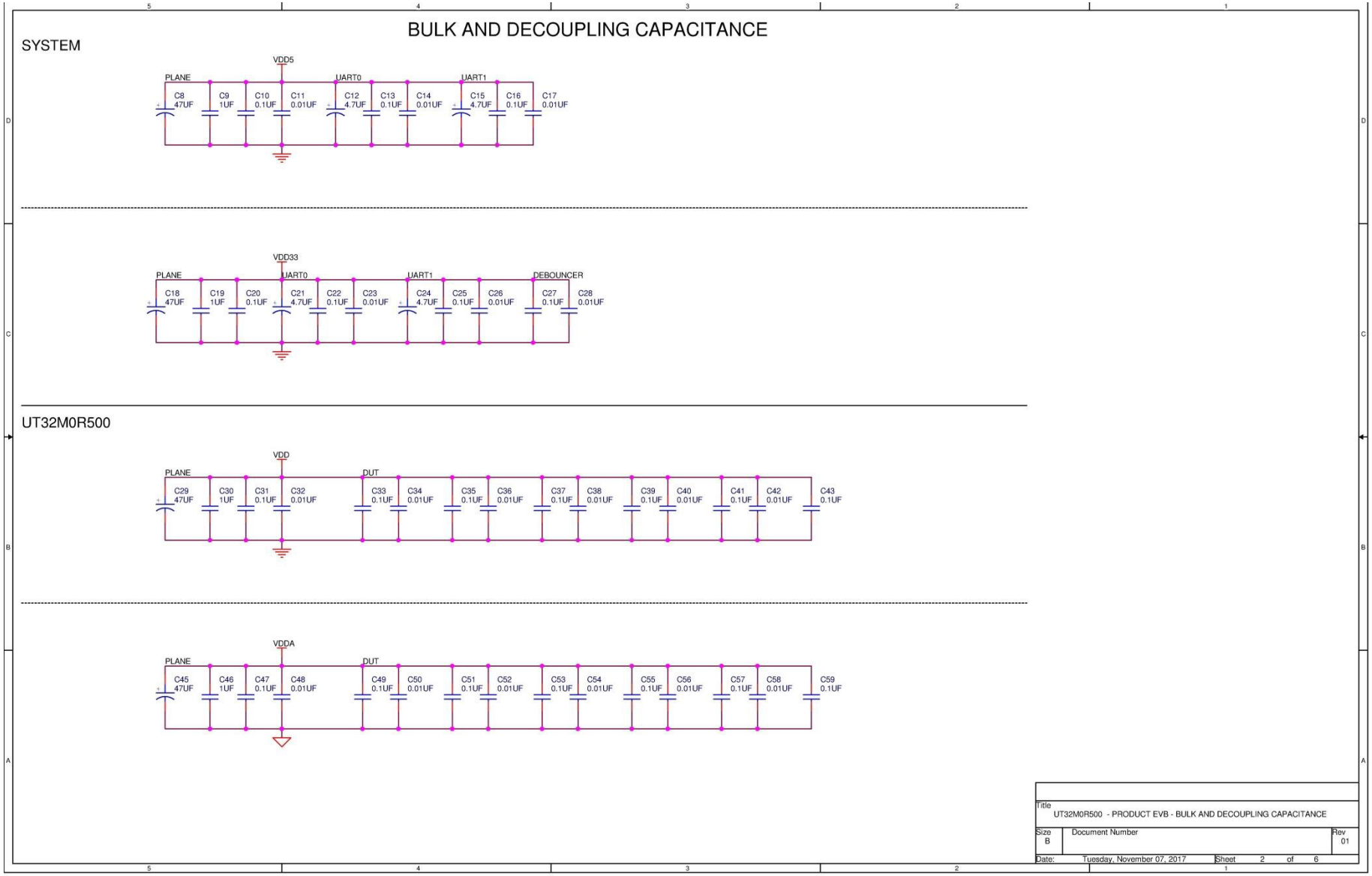
Using the Calculated value, send 'CRC –c<value>'.

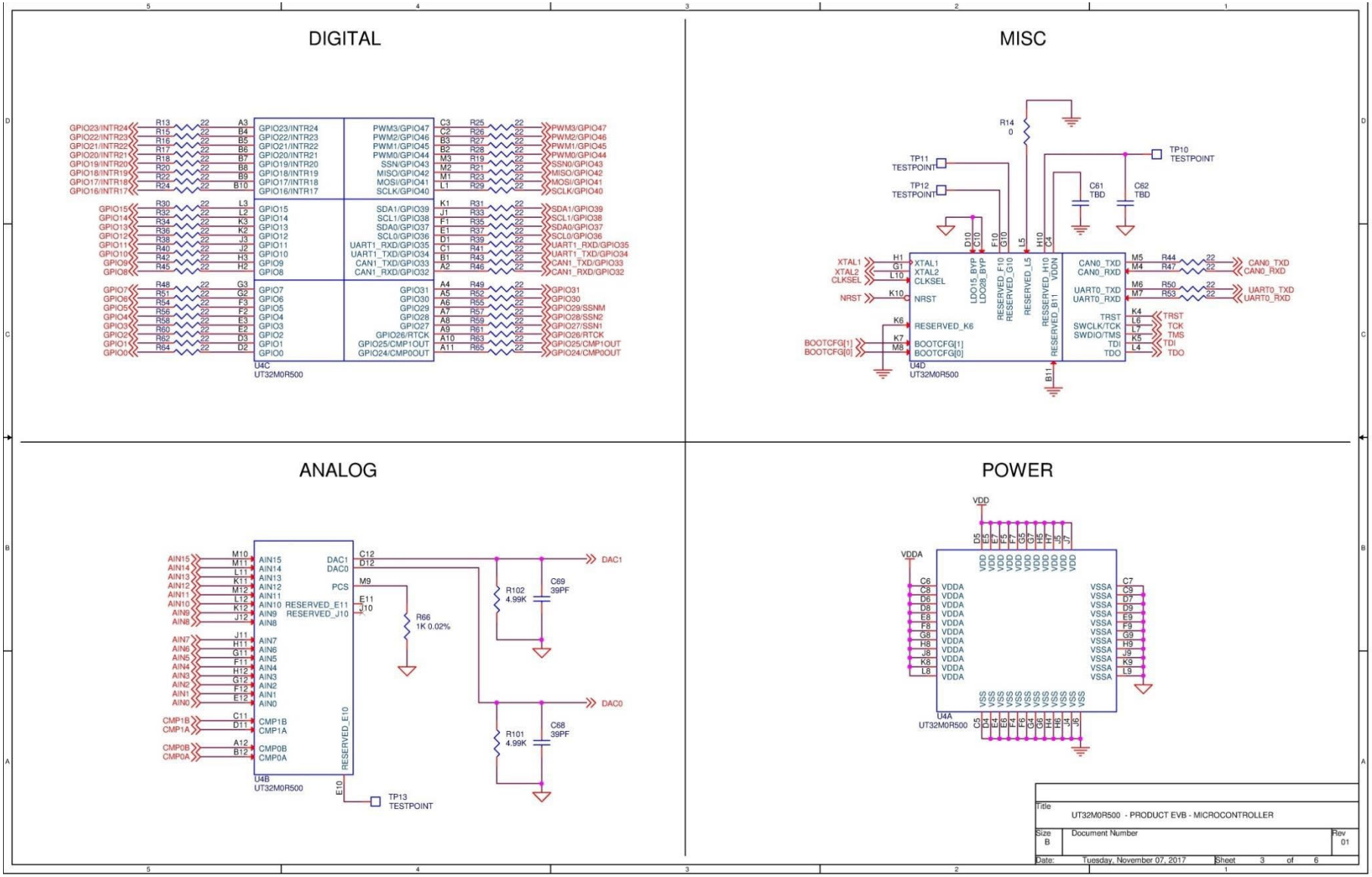
```
:>CRC -cCDF1
SUCCESS!!  CRC programmed correctly
```

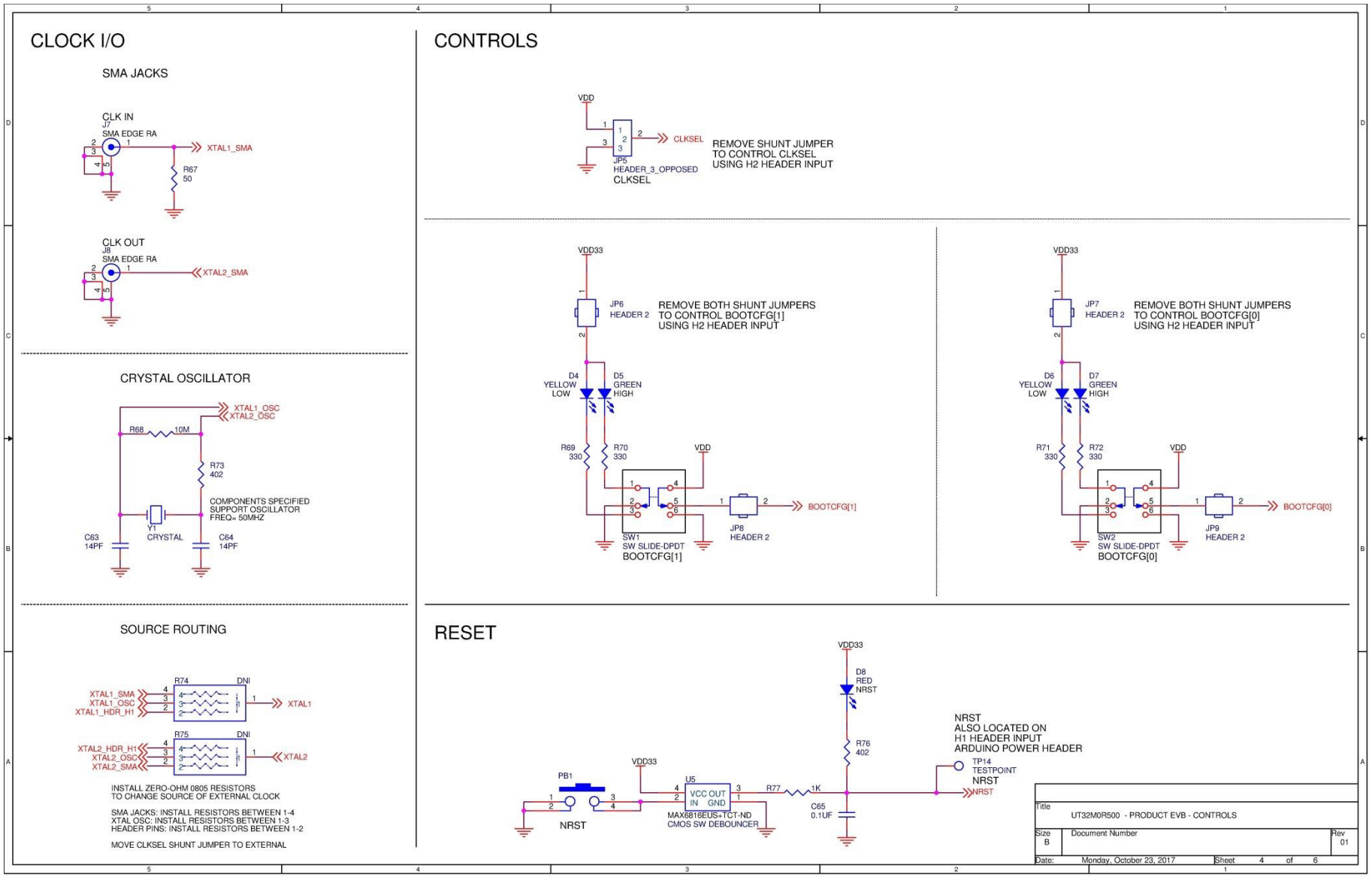
Finally send 'VFY' again. This time, you should see that CRC matches the expected value. You may now change BOOTCFG to b'00 and hit RESET (PB1) to run your program.

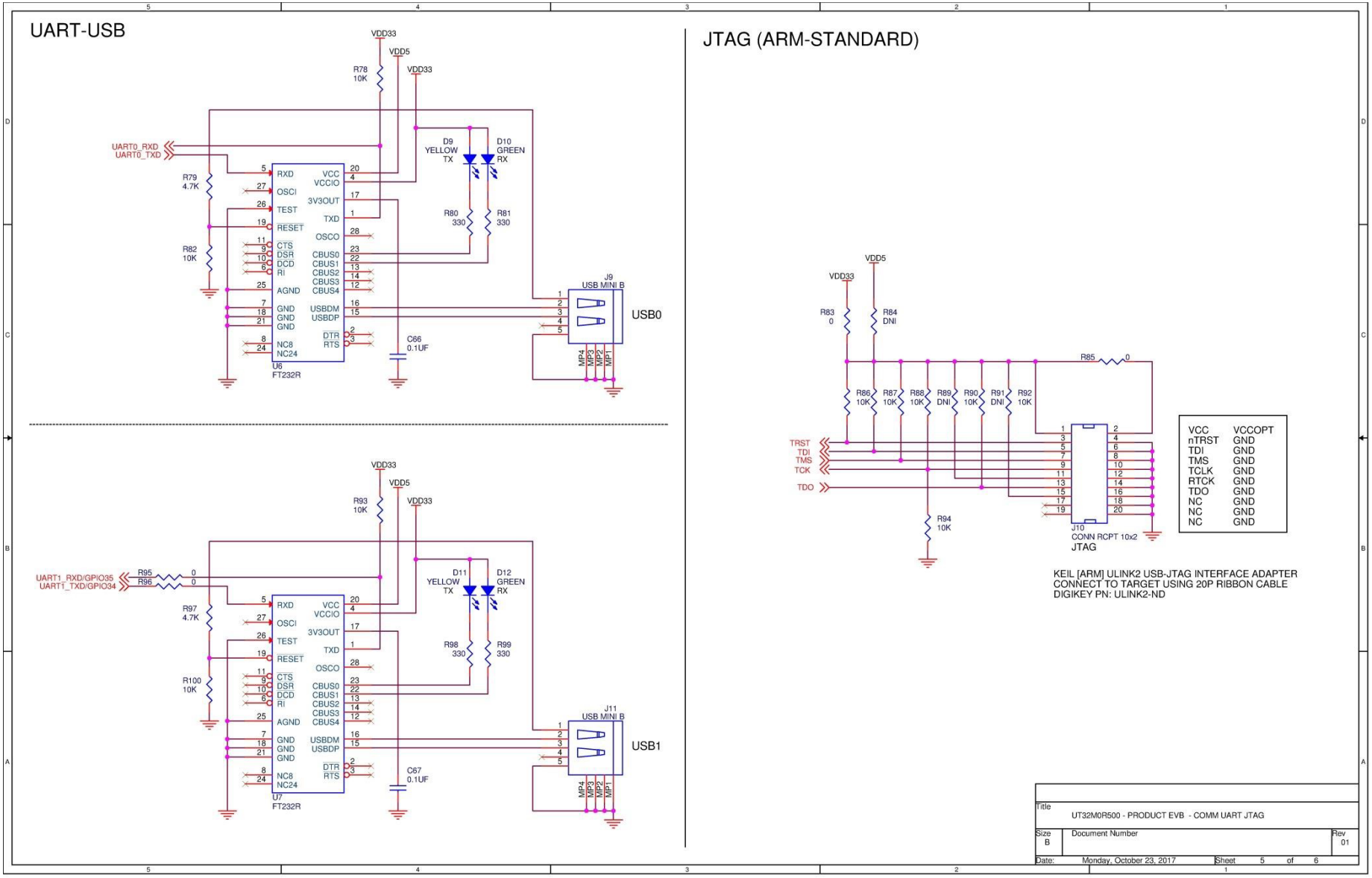
9.0 UT32M0R500-EVB Electrical Schematics

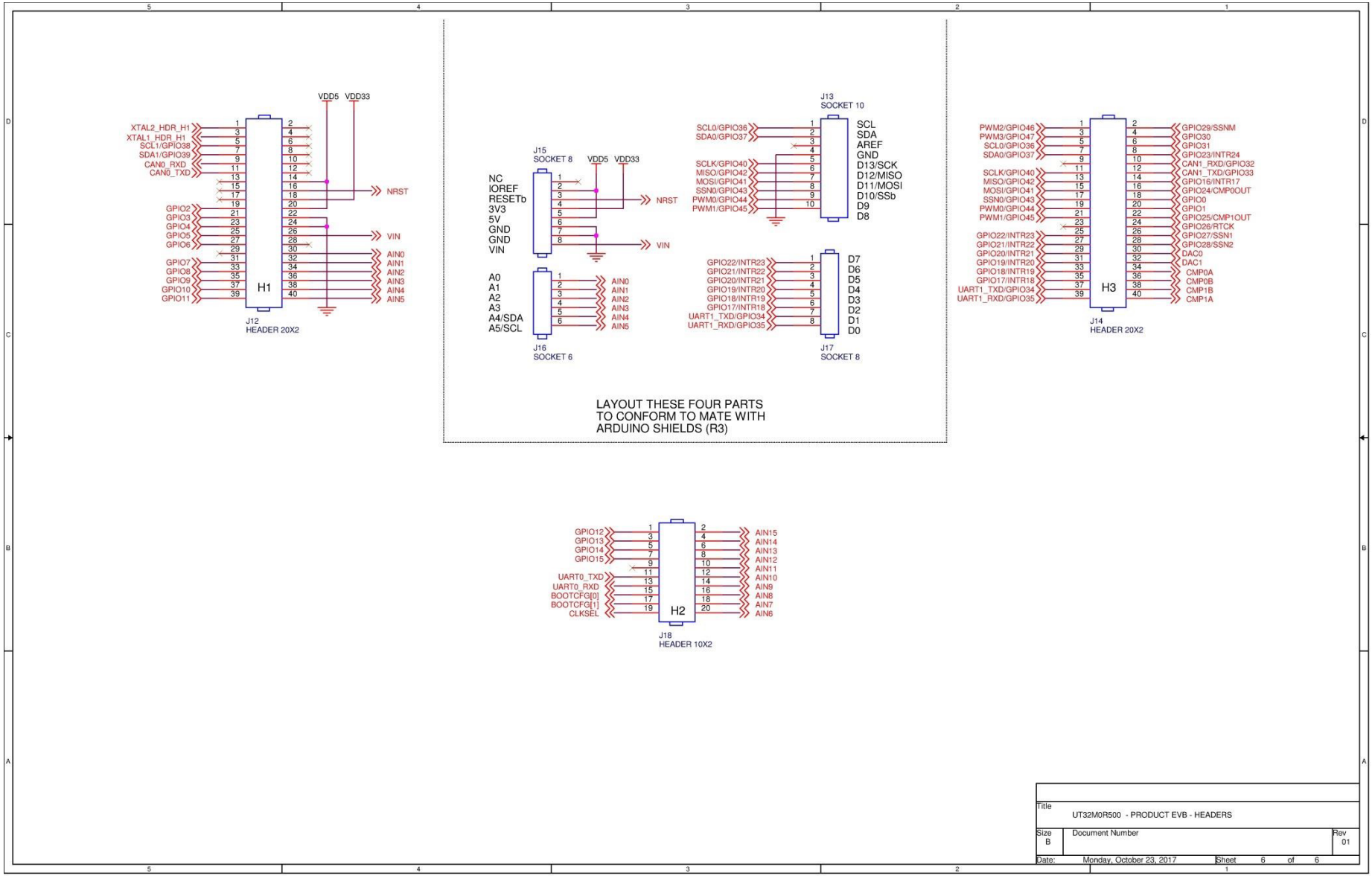



BULK AND DECOUPLING CAPACITANCE
SYSTEM
VDD5
PLANE
UART0
UART1
C8 47UF
C9 1UF
C10 0.1UF
C11 0.01UF
C12 4.7UF
C13 0.1UF
C14 0.01UF
C15 4.7UF
C16 0.1UF
C17 0.01UF
VDD33
PLANE
UART0
UART1
DEBOUNCER
C18 47UF
C19 1UF
C20 0.1UF
C21 4.7UF
C22 0.1UF
C23 0.01UF
C24 4.7UF
C25 0.1UF
C26 0.01UF
C27 0.1UF
C28 0.01UF
UT32M0R500
VDD
PLANE
DUT
C29 47UF
C30 1UF
C31 0.1UF
C32 0.01UF
C33 0.1UF
C34 0.01UF
C35 0.1UF
C36 0.01UF
C37 0.1UF
C38 0.01UF
C39 0.1UF
C40 0.01UF
C41 0.1UF
C42 0.01UF
C43 0.1UF
VDDA
PLANE
DUT
C45 47UF
C46 1UF
C47 0.1UF
C48 0.01UF
C49 0.1UF
C50 0.01UF
C51 0.1UF
C52 0.01UF
C53 0.1UF
C54 0.01UF
C55 0.1UF
C56 0.01UF
C57 0.1UF
C58 0.01UF
C59 0.1UF
Title
UT32M0R500 - PRODUCT EVB - BULK AND DECOUPLING CAPACITANCE
Size B
Document Number
Rev 01
Date: Tuesday, November 07, 2017
Sheet 2 of 6


DIGITAL
MISC
ANALOG
POWER
U4C
UT32M0R500
U4D
UT32M0R500
U4B
UT32M0R500
U4A
UT32M0R500
TP10
TESTPOINT
TP11
TESTPOINT
TP12
TESTPOINT
TP13
TESTPOINT
R14
0
C61
TBD
C62
TBD
R66
1K 0.02%
R102
4.99K
R101
4.99K
C69
39PF
C68
39PF
DAC1
DAC0
VDD
VDDA
Title
UT32M0R500 - PRODUCT EVB - MICROCONTROLLER
Size
B
Document Number
Rev
01
Date:
Tuesday, November 07, 2017
Sheet
3
of
6


CLOCK I/O
SMA JACKS
CLK IN
J7
SMA EDGE RA
XTAL1_SMA
R67
50
CLK OUT
J8
SMA EDGE RA
XTAL2_SMA
CRYSTAL OSCILLATOR
XTAL1_OSC
XTAL2_OSC
R68
10M
R73
402
COMPONENTS SPECIFIED
SUPPORT OSCILLATOR
FREQ= 50MHZ
Y1
CRYSTAL
C63
14PF
C64
14PF
SOURCE ROUTING
R74
DNI
XTAL1_SMA
XTAL1_OSC
XTAL1_HDR_H1
XTAL1
R75
DNI
XTAL2_HDR_H1
XTAL2_OSC
XTAL2_SMA
XTAL2
INSTALL ZERO-OHM 0805 RESISTORS
TO CHANGE SOURCE OF EXTERNAL CLOCK
SMA JACKS: INSTALL RESISTORS BETWEEN 1-4
XTAL OSC: INSTALL RESISTORS BETWEEN 1-3
HEADER PINS: INSTALL RESISTORS BETWEEN 1-2
MOVE CLKSEL SHUNT JUMPER TO EXTERNAL
CONTROLS
VDD
CLKSEL
JP5
HEADER_3_OPPOSED
CLKSEL
REMOVE SHUNT JUMPER
TO CONTROL CLKSEL
USING H2 HEADER INPUT
VDD33
JP6
HEADER 2
REMOVE BOTH SHUNT JUMPERS
TO CONTROL BOOTCFG[1]
USING H2 HEADER INPUT
D4
YELLOW
LOW
D5
GREEN
HIGH
R69
330
R70
330
VDD
SW1
SW SLIDE-DPDT
BOOTCFG[1]
JP8
HEADER 2
BOOTCFG[1]
VDD33
JP7
HEADER 2
REMOVE BOTH SHUNT JUMPERS
TO CONTROL BOOTCFG[0]
USING H2 HEADER INPUT
D6
YELLOW
LOW
D7
GREEN
HIGH
R71
330
R72
330
VDD
SW2
SW SLIDE-DPDT
BOOTCFG[0]
JP9
HEADER 2
BOOTCFG[0]
RESET
VDD33
D8
RED
NRST
R76
402
NRST
ALSO LOCATED ON
H1 HEADER INPUT
ARDUINO POWER HEADER
TP14
TESTPOINT
NRST
NRST
PB1
NRST
VDD33
U5
VCC OUT
IN GND
MAX6816EUS+TCT-ND
CMOS SW DEBOUNCER
R77
1K
C65
0.1UF
Title
UT32M0R500 - PRODUCT EVB - CONTROLS
Size
B
Document Number
Rev
01
Date:
Monday, October 23, 2017
Sheet
4
of
6


UART-USB
JTAG (ARM-STANDARD)
VDD33
VDD5
R78 10K
D9 YELLOW TX
D10 GREEN RX
UART0_RXD
UART0_TXD
R79 4.7K
R80 330
R81 330
R82 10K
RXD
OSCI
TEST
RESET
3V3OUT
TXD
OSCO
CTS
DSR
DCD
RI
AGND
GND
CBUS0
CBUS1
CBUS2
CBUS3
CBUS4
USBDM
USBDP
NC8
NC24
U6
FT232R
C66 0.1UF
J9
USB MINI B
USB0
R83 0
R84 DNI
R85 0
R86 10K
R87 10K
R88 10K
R89 DNI
R90 10K
R91 DNI
R92 10K
R94 10K
TRST
TDI
TMS
TCK
TDO
J10
CONN RCPT 10x2
JTAG
VCC VCCOPT
nTRST GND
TDI GND
TMS GND
TCLK GND
RTCK GND
TDO GND
NC GND
NC GND
NC GND
KEIL [ARM] ULINK2 USB-JTAG INTERFACE ADAPTER
CONNECT TO TARGET USING 20P RIBBON CABLE
DIGIKEY PN: ULINK2-ND
R93 10K
D11 YELLOW TX
D12 GREEN RX
UART1_RXD/GPIO35
UART1_TXD/GPIO34
R95 0
R96 0
R97 4.7K
R98 330
R99 330
R100 10K
U7
FT232R
C67 0.1UF
J11
USB MINI B
USB1
Title
UT32M0R500 - PRODUCT EVB - COMM UART JTAG
Size B
Document Number
Rev 01
Date: Monday, October 23, 2017
Sheet 5 of 6


VDD5
VDD33
XTAL2_HDR_H1
XTAL1_HDR_H1
SCL1/GPIO38
SDA1/GPIO39
CAN0_RXD
CAN0_TXD
NRST
GPIO2
GPIO3
GPIO4
GPIO5
GPIO6
VIN
GPIO7
GPIO8
GPIO9
GPIO10
GPIO11
AIN0
AIN1
AIN2
AIN3
AIN4
AIN5
H1
J12
HEADER 20X2
J15
SOCKET 8
NC
IOREF
RESETb
3V3
5V
GND
GND
VIN
J16
SOCKET 6
A0
A1
A2
A3
A4/SDA
A5/SCL
J13
SOCKET 10
SCL0/GPIO36
SDA0/GPIO37
SCLK/GPIO40
MISO/GPIO42
MOSI/GPIO41
SSN0/GPIO43
PWM0/GPIO44
PWM1/GPIO45
SCL
SDA
AREF
GND
D13/SCK
D12/MISO
D11/MOSI
D10/SSb
D9
D8
J17
SOCKET 8
GPIO22/INTR23
GPIO21/INTR22
GPIO20/INTR21
GPIO19/INTR20
GPIO18/INTR19
GPIO17/INTR18
UART1_TXD/GPIO34
UART1_RXD/GPIO35
D7
D6
D5
D4
D3
D2
D1
D0
LAYOUT THESE FOUR PARTS
TO CONFORM TO MATE WITH
ARDUINO SHIELDS (R3)
PWM2/GPIO46
PWM3/GPIO47
SCL0/GPIO36
SDA0/GPIO37
SCLK/GPIO40
MISO/GPIO42
MOSI/GPIO41
SSN0/GPIO43
PWM0/GPIO44
PWM1/GPIO45
GPIO22/INTR23
GPIO21/INTR22
GPIO20/INTR21
GPIO19/INTR20
GPIO18/INTR19
GPIO17/INTR18
UART1_TXD/GPIO34
UART1_RXD/GPIO35
GPIO29/SSNM
GPIO30
GPIO31
GPIO23/INTR24
CAN1_RXD/GPIO32
CAN1_TXD/GPIO33
GPIO16/INTR17
GPIO24/CMP0OUT
GPIO0
GPIO1
GPIO25/CMP1OUT
GPIO26/RTCK
GPIO27/SSN1
GPIO28/SSN2
DAC0
DAC1
CMP0A
CMP0B
CMP1B
CMP1A
H3
J14
HEADER 20X2
GPIO12
GPIO13
GPIO14
GPIO15
UART0_TXD
UART0_RXD
BOOTCFG[0]
BOOTCFG[1]
CLKSEL
AIN15
AIN14
AIN13
AIN12
AIN11
AIN10
AIN9
AIN8
AIN7
AIN6
H2
J18
HEADER 10X2
Title
UT32M0R500 - PRODUCT EVB - HEADERS
Size
B
Document Number
Rev
01
Date:
Monday, October 23, 2017
Sheet
6
of
6

Revision History

Date	Revision #	Author	Change Description	Page #
11/2017	0.0.1	OW/AW	DRAFT	
12/2017	0.0.2	OW/AW	DRAFT REVISON	
12/2017	0.0.3/4	OW/AW	Added information on setting up a project and running a sample program. Added information on how to program over UART.	
12/2017	0.1.0	OW/AW	Draft release.	
3/2018	1.0.0	OW/AW/JA	Initial Release	
5/20/2021	1.1.0	OW	Updated template; Added the remaining 'Target 1' generic settings pictures, including the JTAG/Cortex M setup	

Frontgrade Technologies Proprietary Information Frontgrade Technologies (Frontgrade or Company) reserves the right to make changes to any products and services described herein at any time without notice. Consult a Frontgrade sales representative to verify that the information contained herein is current before using the product described herein. Frontgrade does not assume any responsibility or liability arising out of the application or use of any product or service described herein, except as expressly agreed to in writing by the Company; nor does the purchase, lease, or use of a product or service convey a license to any patents, rights, copyrights, trademark rights, or any other intellectual property rights of the Company or any third party.