HI-6131
Application Development Kit

July 2016

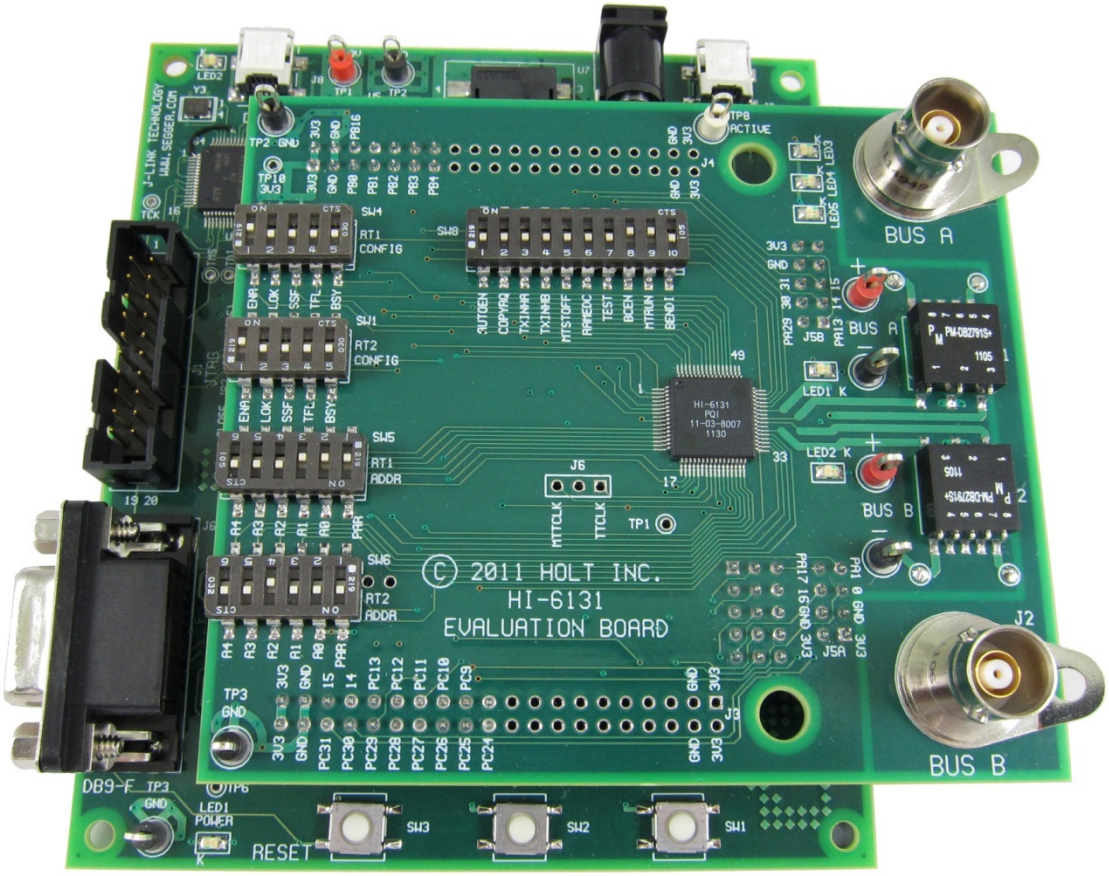
REVISION HISTORY

Revision	Date	Description of Change
AN-6131, Rev. New	10/24/11	Initial Release
Rev. A	03/12/12	Replaced HI-6130 schematic with HI-6131 schematic
Rev. B	06/15/12	Revise the document and refresh the software project. The new project will only support the HI-6131 (not HI-6130 anymore) but provide an organized project using folders and configurations in the IAR IDE.
Rev. C	07/01/12	Clarify software installation process.

Introduction

The Holt HI-6131 Application Development Kit demonstrates the broad feature set of the HI-6131 Multi Terminal IC for MIL-STD-1553. The 2-board assembly and C project reference design provides a ready-to-run evaluation platform demonstrating concurrent operation for any combination of Bus Controller, Bus Monitor and one or two Remote Terminals. For convenience, this kit includes IAR Systems Embedded Workbench® for ARM, and a fully integrated debug interface for the ARM Cortex M3 microcontroller.

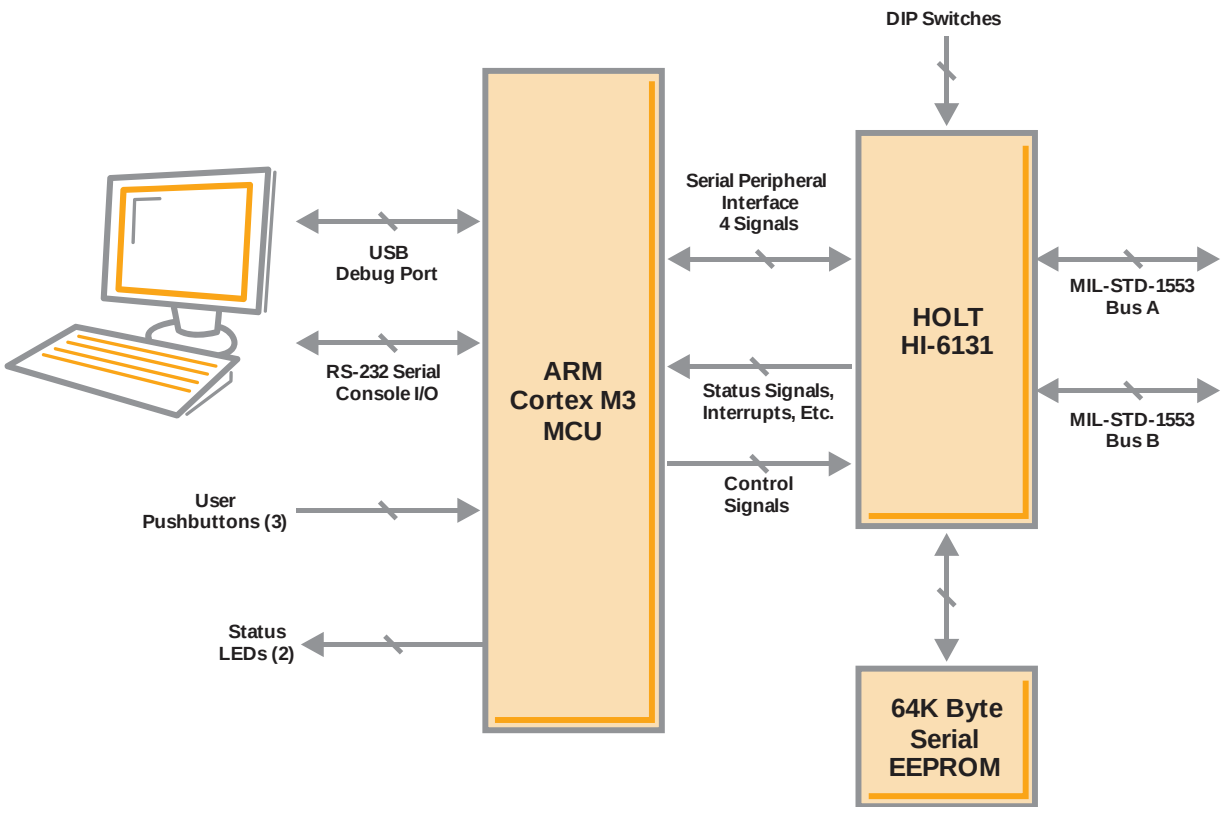
This guide describes how to set up and run the board. Additional support material and all required project software are found in the included Holt CD-ROM. A version of the demonstration software is already programmed into the microcontroller flash; the board is operational right out of the box without installing or running the provided software development tools.



Evaluation Kit Contents

- This User Guide.
- Holt HI-6131 Project Software and Documentation CD.
- Installation CD for IAR Systems Embedded Workbench® for ARM (32KB KickStart ed.).
- Plug-in DC power supply.
- USB debug interface cable.
- RS-232 serial cable, DB-9M to DB-9F for console I/O using a connected computer.
- 2-board assembly comprised of
    - Upper HI-6131 board with dual transformer-coupled MIL-STD-1553 bus interfaces. Numerous DIP switches configure board operation.
    - Lower MCU board with ARM Cortex M3 16-/32-bit microprocessor, debug interface and regulated 3.3VDC power supply.

Hardware Block Diagram



Hardware Design Overview

Refer to the end of this guide for separate schematic diagrams and bills of material for the upper and lower circuit boards.

The detachable HI-6131 board can be separated from the provided MCU board for connection to a user-supplied alternate microprocessor or FPGA board. The inter-board headers are located on 0.1" (2.54 mm) grid for compatibility with generic prototyping boards. All host interface signals go through the inter-board headers. Numerous HI-6131 configuration pins (and the Remote Terminal address setting pins) are controlled by DIP switches on the upper HI-6131board; these signals are not available on the inter-board headers.

The lower ARM Cortex M3 board is based on the flash-programmable Atmel AT91SAM3U-EK microprocessor. A 4-signal Serial Peripheral Interface(SPI) connects to the HI-6131. A UART-based serial port provides RS-232 console I/O (optional). An uncommitted USB 2.0 port is available for future expansion. Two pushbuttons are available for software interaction. A RESET pushbutton resets the ARM microprocessor, which in turn controls the HI-6131 Master Reset signal.

The ARM Cortex M3 board includes "J-Link On Board" debug interface, licensed from www.segger.com, providing out-of-box readiness without having to buy a costly JTAG debug cable. The kit includes a simple USB cable for connecting the board's debug interface to your computer. (For users already owning an ARM debug interface with ribbon-cable connector, an ARM-standard 2x10 debug connector provides debug connectivity. In this case, jumper JP2 on the bottom of the lower board should be soldered closed to disable "J-Link On Board".)

A Quick Demonstration

The Holt HI-6131Application Development Kit is pre-programmed to concurrently operate as a Bus Controller, SMT Bus Monitor and two independent Remote Terminals. Terminal addresses for the two RTs are preset using DIP switches, before applying power. RT addresses 3 and 4 are utilized by the preprogrammed Bus Controller message repertoire. The two 6-position DIP switches should already be set with these address values, plus odd parity.

1. To observe bus activity, connect an oscilloscope to the red BUS A and red BUS B test points. The test point labeled ACTIVE is a convenient scope trigger signal.

2. If not connected by cable to MIL-STD-1553 buses, provide a dummy load for buses A and B by connecting a 70Ω 1/2 Watt resistor across each pair of red and black Bus test points. (For this demonstration, half-Watt resistors are adequate because duty cycle is sufficiently low.)

3. The preprogrammed demonstration provides console I/O between the MCU board serial port and a Windows computer running a terminal emulation program. Using the provided DB-9 serial cable, connect the MCU board to the serial port on the Windows computer. . If using Vista or Windows 7, install TeraTerm (see page 5) and run it. If using XP or Windows 2000, you may instead open HyperTerminal from the Windows "Accessories" program group, although TeraTerm is recommended. Configure the TeraTerm program for serial port (not TCP I/P), 115,200 baud, no parity, 8 data bits, 1 stop bit, flow control off. If configured correctly, a text header appears when power is applied, or when the RESET pushbutton is pressed.

4. Plug-in the provided 5V DC power supply and connect the cable to the power input jack on the lower circuit board. The command menu should appear in the TeraTerm window.

5. The Bus Controller is programmed to execute a repeating series of MIL-STD-1553 commands to Remote Terminal addresses 3 and 4. Each bus command is preceded by a BC "Wait for Trigger" op code. The MCU is preprogrammed to issue a trigger pulse to the BC each time the numeric "1" key is pressed on the computer keyboard. (By turning off the console I/O option and recompiling the program, the Bus Controller can be paced instead by pressing the SW1 button on the MCU board, but the TeraTerm console output will be disabled in this mode.)

6. Each time the numeric "1" key is pressed on the computer keyboard, a new Bus Controller command is issued to RT address 3 or RT address 4 (or both RT addresses 3 and 4, in the case of RT-RT messages). The Remote Terminal responses can be observed, and the TeraTerm console screen reports new message results for each key press.

Getting Started With Full Evaluation

The following steps install and configure the C compiler and describe how to load and modify demonstration projects using the HI-6131Application Development Kit.

1. Embedded Workbench for ARM is a fully functional integrated development environment including project manager, editor, compiler, assembler, linker, librarian and debugger tools. It includes an optimizing C compiler, and supports a wide range of ARM devices and hardware debug systems. Ready-made device configuration files, flash loaders and example projects are included.

2. The installation guide "Demo Project Installation for IAR Systems Embedded Workbench for ARM" included in the project folder contains steps how to install IAR and what folder the Holt project should be unzipped to. If your compiled program exceeds 32K bytes, try disabling the console I/O option (in project file **`613x_config.h`**).This significantly reduces compiled program size without compromising MIL-STD-1553 functionality. Otherwise, a 30 day evaluation license for the unrestricted version of Embedded Workbench for ARM should be used.

3. If using console I/O (recommended for initial evaluation, your computer needs a serial (COM) port and a "terminal emulation" program like TeraTerm. Most desktop computers have a COM port; many notebook computers do not have a COM port; these require a serial-to-USB adapter.

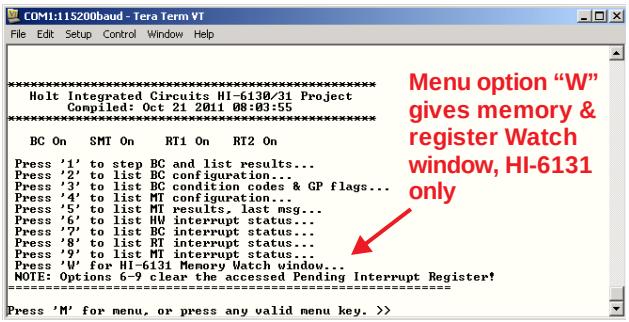
   If using Windows 2000 or Windows XP, you can use HyperTerminal for terminal emulation. Open HyperTerminal by clicking **Start** then **All Programs;** click the Windows **Accessories** then **Communications** program group. Double-click HyperTerminal to run it. Skip the next paragraph.

   If using Vista or Windows 7…
   HyperTerminal is not included with these versions of Windows. Install the free open-source terminal emulation program, *TeraTerm 4.71*, by running the provided teraterm-4.71.exe installer program from the Holt CD. Accept the license agreement stating redistribution is permitted provided that copyright notice is retained. The notice can be displayed from the TeraTerm window by clicking **Help** then clicking **About TeraTerm**. Continuing to install…

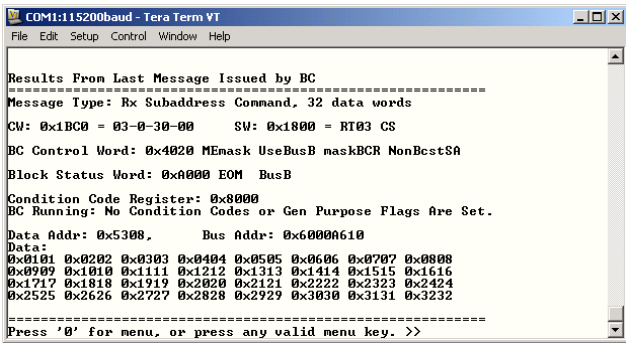
   - Accept the default install destination and click **Next**.
   - At the Select Components screen, unselect all options except Additional Plugin = TTXResizeMenu and click **Next**.
   - Select the installed language, then click **Next**.
   - Accept the default Start Menu folder, then click **Next**.
   - Select any desired shortcuts, then click **Next**.
   - At the Ready to Install screen, click **Install**.

   Run the TeraTerm program. At the **New Connection** screen, select **(x)Serial** and choose the selected COM port. Click **Setup** then **Serial Port** to open the serial port setup window. Choose these settings: Baud Rate: 115200, Data: 8 bits, Parity: none, Stop: 1 bit, Flow Control: none
   Using the provided DB-9 serial cable, connect the MCU board to the computer serial (COM) port.

4. Plug-in the provided 5V DC power supply and connect the cable to the power input jack on the lower circuit board. If TeraTerm is running and configured correctly, the command menu below should appear in the console window. This menu appears whenever board power is applied, or when the RESET pushbutton is pressed. After verifying correct TeraTerm communication with the evaluation board, the terminal set up can be saved by clicking **Setup** then **Save Setup**.



Fewer menu options are listed if terminal devices (BC, RT, MT) are disabled. Here is an example console screen, after pressing '1' on the keyboard from the previous menu:



5. Debug requires an interface between the computer running IAR Embedded Workbench® and the HI-6131Application Development Kit. Connect the small end of the provided USB cable to the HI-6131 evaluation board USB connector marked DEBUG. Connect the other end of the USB cable to a free computer USB port. The IAR C-SPY Debugger for ARM includes drivers for numerous target system interfaces, including built in "J-link On Board".

   The first time the evaluation board USB cable is connected to the computer, the Windows "Found New Hardware" message should appear for the J-Link device. After several seconds, Windows should load the appropriate driver and advise, "Your hardware is ready for use". If Windows fails to find the J-Link driver, direct it to look in the Drivers directory the IAR Embedded Workbench® installation CD.

   If difficulties arise when initiating a debug session at step 11, click **Project** then **Options**. In the window that opens, under **Category** = **Debugger** highlight **J-Link/J-Trace**. Click the tab labeled **Connection**, then verify Communications = USB and Interface = SWD.

6. Open IAR Embedded Workbench®. Click **File**, then **Open Workspace**, then navigate to the project subdirectory created in step 5. Select the project file with .EWW extension, then click **Open**. (The next time Embedded Workbench® opens, this project will appear in the Recent Workspaces list when **File** is clicked.)

7. The HI-6131 project only uses unsigned integer variables. Turn off the nuisance compiler message that occurs when a variable's most significant bit toggles. The message looks like this:
   `Remark[Pe068]: integer conversion resulted in a change of sign`
   To disable this diagnostic message, click **Project** then click **Options**
   Category = C/C++ Compiler
   Tab = Diagnostics
   `Suppress these diagnostics:` add "Pe068" to list

8. The Holt IAR project includes seven predefined configurations which can be selected from the workspace pull-down menu. The configurations modify the preprocessor labels BC_ena, RT1_ena, RT2_ena, SMT_ena and IMT_ena. These are compile time labels used by the compiler to customize the builds.

   The default configuration BC_MT_RT enables the primary modes BC, MT, RT1 and RT2 in the HI-613X device. This is the default configuration programmed into the demo board. This enables and demonstrates the Bus Controller, RT1 and RT2 Remote Terminals and a Monitor. These configurations are all flash based projects. RAM based projects are not supported due to the limited amount of RAM on the MCU. By design the Cortex™-M3 runs slower in RAM than in Flash so there is little need for a RAM based project. The seven configurations and the corresponding preprocessor label values are provided in the table below:

Configuration	BC_ena	RT1_ena	RT2_ena	SMT_ena	IMT_ena
BC_MT_RT (default)	1	1	1	1	0
BC_ONLY	1	0	0	0	0
SMT_ONLY	0	0	0	1	0
RT_ONLY	0	1	1	0	0
IMT_ONLY	0	0	0	1	0
RT_SMT	0	1	1	1	0
RT_IMT	0	1	1	0	1

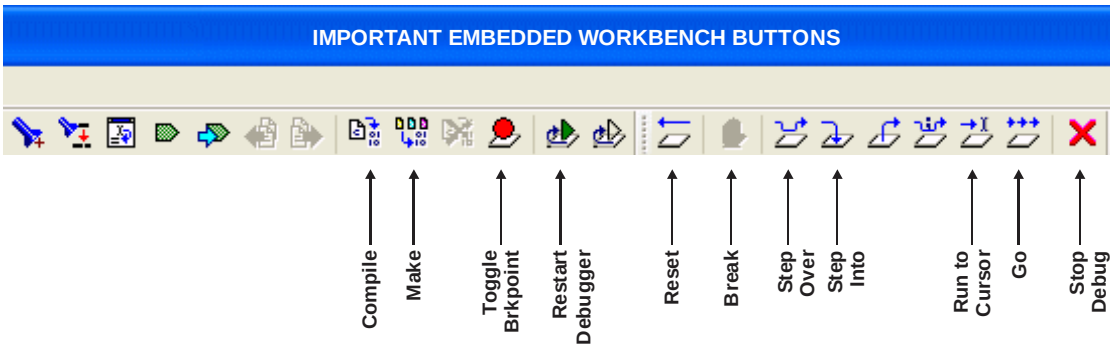
   Other configurations are possible except just one type of Monitor can be selected at a time, so SMT_ena and IMT_ena cannot both be enabled (set to 1). A simple way to create a new configuration is to select Project/Edit Configurations and then select New. The dialog box will allow a new configuration based on an existing configuration with a new configuration name. Select the new configuration and edit the preprocessor labels as desired then save the new configuration. The new configuration will now appear in the pull down menu.

   When changing the configurations from the default the corresponding four terminal DIP switches on the board BCENA, MTRUN, RT1ENA and RT2ENA should be set to match. If any of these do not match an error will be displayed on the console when the board first powers up.

9. Project file 613x_config.h configures other critical project settings, including the time tag resolution and console I/O on-off.

10. Compile the project by clicking the **Make** button. See following illustration. If the Build messages window in IAR Embedded Workbench® indicates no errors or warnings, you can continue. If errors occurred, correct them and recompile the program.

11. Initiate a debug session by clicking the **Restart Debugger** button. This downloads the compiled program into the MCU and readies the board for program execution. Click **Go** to start execution. Click **Break** (normally displayed during execution as a red upheld hand) to stop execution.



12. To program the auto-initialization serial EEPROM with a new configuration:
    - Before starting program execution, turn off the DIP switch labeled AUTOEN, directing the MCU to initialize the HI-6131 instead of using self-initialization from the EEPROM. Turn on the DIP switch labeled COPYREQ, directing the MCU to initiate the EEPROM copy sequence after post-reset initialization of HI-6131 registers and RAM is complete.
    - When execution starts, the red LED illuminates during the EEPROM copy process. When it turns off, turn on the DIP switch labeled AUTOEN to reactivate self-initialization. Turn off the COPYREQ DIP switch, preventing EEPROM rewrite at each reset.

13. To observe bus activity, connect an oscilloscope to the red BUS A and red BUS B test points. The test point labeled ACTIVE is a convenient scope trigger signal. If not connected by cable to MIL-STD-1553 buses, provide a dummy loads for buses A and B by connecting a 70Ω 1 Watt resistor across each pair of red and black Bus test points.

Project File List with Selected Descriptions

HEADER FILES WITHOUT CORRESPONDING C FILES

device_6131.h

ONLY USED FOR HI-6131 (SPI) PROJECTS

Macro definitions for HI-6131 register addressing

613x_initialization.h

Definitions for important configuration settings

613x_regs.h

Macros for register bits and bit fields

C FILES WITH CORRESPONDING HEADER FILES

Most of the function names are self-explanatory.

main.c

`main();`

The primary program entry portal, `main()` demonstrates the initialization sequence used, whether or not self-initialization from EEPROM is enabled, for any combination of enabled terminals. After initialization is complete, function calls demonstrate powerful addressing methods for all RAM structures used by the enabled terminal modes.

board_613x.c

`board_613x.h contains ARM MCU i/o definitions`

`ConfigureGpio();` initializes ARM MCU general purpose I/O

```
reset_613x();
autoinit_check();
initialize_613x_shared();
init_timer();
Delay_us(num_us);
Delay_ms(num_ms);
Delay_x100ms(num);
Flash_Red_LED();
Flash_Green_LED();
error_trap(count);
enable_check();
write_init_eeprom();
```

board_6131.c

ONLY USED FOR HI-6131 (SPI) PROJECTS

```
board_6131.h contains ARM MCU SPI i/o definitions and macro
definitions for SPI commands

SPIopcode(opcode) ;
Write_6131LowReg(reg_number, data, irq_mgmt) ;
Read_6131LowReg(reg_number, irq_mgmt) ;
Write_6131_1word(data, irq_mgmt) ;
Read_6131_1word(irq_mgmt) ;
Write_6131(write_data[], inc_pointer_first, irq_mgmt) ;
Read_6131(number_of_words, irq_mgmt) ;
Write_6131_Buffer(write_data[], inc_pointer_first, irq_mgmt) ;
Read_6131_Buffer(number_of_words, inc_pointer_first, irq_mgmt) ;
Read_Current_Control_Word(rt_num, irq_mgmt) ;
getMAPaddr() ;
enaMAP(map_num) ;
Read_Current_Control_Word(rt_num, irq_mgmt);
Read_RT1_Control_Word(txrx, samc, number, irq_mgmt);
Read_RT2_Control_Word(txrx, samc, number, irq_mgmt);
ReadWord_Adv4(irq_mgmt) ;
Read_Last_Interrupt(irq_mgmt) ;
Fill_6131RAM_Offset() ;
Fill_6131RAM(addr, num_words, fill_value) ;
Memory_watch(address);
Configure_ARM_MCU_SPI();
```

613x_BC.c

```
613x_BC.h contains bus addressing structs used only by HI-6130
613x_BC.h has instruction list macros used HI-6130 or HI-6131

BC_bus_addressing_examples();(HI-6130 only)
initialize_bc_msg_blocks();
initialize_bc_instruction_list();
initialize_613x_BC();
bc_disable();
bc_enable();
bc_start();
bc_trigger();
bc_switch_tests();
```

For the demo, this function polls pushbutton SW1 and triggers next BC message when pressed

```
SW1_BC_Trigger();
SW2_BCtest ();
initialize_613x_BC();
```

613x_MT.c

```
613x_MT.h contains bus addressing structs (HI-6130 only)
IMT_bus_addressing_examples(); (HI-6130 only)
SMT_bus_addressing_examples(); (HI-6130 only)
initialize_613x_MT();
```

This function initializes either simple or IRIG-106 (SMT or IMT) monitor operation

613x_RT.c

```
613x_RT.h contains bus addressing structs used only by HI-6130
613x_RT.h has descriptor table address macros used by HI-6131

RT_bus_addressing_examples(); (HI-6130 only)
initialize_613x_RT1();
initialize_613x_RT2();
RTAddr_okay(RTnum);
modify_RT_status_bits();
RTstatusUpdate();
```

This function updates RT status bits based on DIP switch settings

```
write_dummy_tx_data_RT1();
write_dummy_tx_data_RT2();
```

The last two functions initialize RT1 and RT2 transmit data buffersfor the demo

console.c

Console functions used by all terminal modes:

```
ConfigureUsart1();
text_header();
chk_key_input();
list_hw_ints_console();
```

Console functions used by Bus Controller (BC) mode:

```
bc_last_msg_console();
list_bc_config ();
list_bc_ccgpf_reg();
list_bc_ints_console();
```

Console functions used by Remote Terminal RT1 and/or RT2:

```
list_rt_ints_console();
```

Console functions used by SMT or IMT bus monitor modes:

```
list_mt_config();
mt_last_msg_console();
list_mt_ints_console();
```

Primitive console functions that "printf" redundant char strings to reduce program size:

```
print_null();
print_sp1sp();
print_b1sp();
print_b0sp();
print_dddn();
print_dd0n();
print_dd1n();
print_menuprompt();
print_line();
```

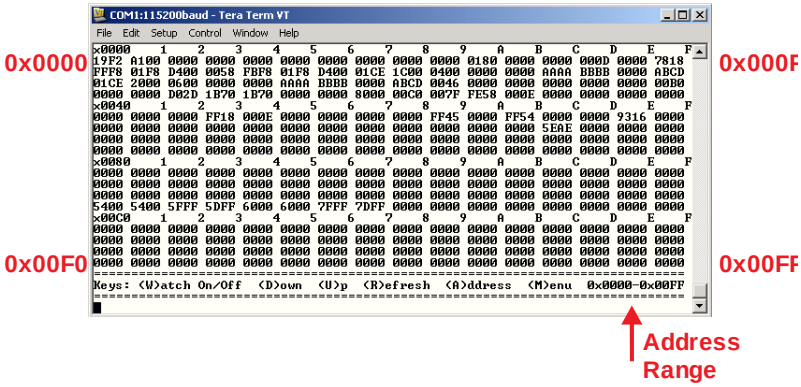
Console function called by HI-6131 Memory_watch() function

```
ascii2int();
```

Application Development Kit Notes

The HI-6131 was designed for compatibility with microcontrollers having a Serial Peripheral Interface (SPI).RAM and register locations are written or read with the help of 8-bit SPI commands. Most read or write operations use one of four Memory Address Pointers (MAPs) to designate the address of the next location accessed. To speed up a multiword transfers, the enabled Memory Address Pointer automatically increments to the next address after each read or write is performed. Register addresses 0-15 decimal can be read directly, without using a memory address pointer. Register addresses 0-63 decimal can be written directly without using a memory address pointer.

When debugging, a memory watch utility may be helpful for observing register or RAM values. The IAR Embedded Workbench debugger includes a powerful Watch window, but this tool only works for memory-mapped features (like HI-6130); it will not work with the SPI interfaced HI-6131. The demonstration program provides similar capability via SPI, by using a C function called Memory_watch(). This function call only works when Console I/O is enabled. It displays 256 consecutive register or RAM values, starting with the provided memory address parameter. The entire memory address space 0 to 0x7FFF is accessible in 256 word increments. The demonstration program polls for keyboard input, and must be running. When the console menu “W” command is entered, the memory address space from 0x0000 to 0x00FF is displayed:



The sub-menu at the bottom of the screen lists available Memory Watch options. Pressing”D” (DOWN command) changes the displayed address range to 0x0100-0x01FF. Pressing”U” (UP command) from the above screen wraps around the device address space, changing the displayed address range to 0x7F00-0x7FFF. Repeating UP or DOWN commands moves through the address range. Pressing “R” refreshes the currently selected address range, while pressing “A” (ADDRESS command) allows you to enter four hexadecimal characters to select any Memory Watch start address. Pressing “W” (WATCH) or “M” (MENU) toggles off Memory Watch display, restoring the menu shown on page 5.

Be mindful that each displayed location is rescanned when Memory_watch( ) executes. Some register or RAM structure bits automatically reset after read occurs. This includes bits in the Pending Interrupt registers, and DBAC Data Block Accessed bits for RT Descriptor Table Control Words in RAM. For these, the Memory Watch window reflects the value in effect when the function executed.

The console I/O option using TeraTerm includes several menu options that read and display Pending Interrupt register status. Remember that Pending Interrupt bits automatically reset after read occurs. For these registers, the Memory Watch window reflects the value in effect when execution stopped.

The HI-6131 demonstration program is set up with Bus Controller, Remote Terminal 1, Remote Terminal 2 and Simple Monitor all enabled. Enabling or disabling any of these terminal functions is a two-step process: the software configuration (controlled by file **`613x_initialization.h`**) must match the hardware configuration DIP switches (BCENA, RT1ENA, RT2ENA and MTRUN)or a software error trap occurs.

The HI-6131 circuit board has two separate serial EEPROMs for auto-initialization. Selectable by a small slide switch SW2 on the upper board (may be on board's bottom side), the DEMO EEPROM is preprogrammed, while the USER EEPROM is shipped unprogrammed. By using SW2 to select the USER EEPROM, experimental project builds can be tried without overwriting the demonstration program in the DEMO EEPROM.

When debugging the IRIG-106 (IMT) Bus Monitor, at times it may be advantageous to disable the Time Tag clock to prevent packet finalization caused by attainment of "maximum recording time." For example, the BC issues a new MIL-STD-1553 command. Using a debugger Memory or Watch window, you wish to examine stored message data before triggering the BC to issue the next command. With the time tag clock running, there is no way a user can examine stored message data without exceeding maximum packet recording time. With time tag clock disabled, a series of messages can be individually issued and checked without packet finalization.

Choosing HI-6130 (Parallel Bus) or HI-6131 (SPI)

The trade-offs…
The HI-6130 features a 16-bit parallel bus interface to the host MCU or FPGA. Its 100-pin plastic QFP package is 16 mm x 16 mm. The HI-6130 signal interface, including bus control signals and chip select, totals 36 pins. The external bus interface in the companion MCU or FPGA also increases its pin count and package size, and increases signal routing area on the circuit board.

In comparison, the HI-6131 features a four wire Serial Peripheral Interface (SPI) to the host MCU or FPGA. It is offered in a 12 mm x 12 mm plastic QFP, or 9 mm x 9 mm chip scale package. Having 32 fewer signals to connect, circuit board area is smaller than the area needed for the HI-6130. If device selection is based solely on pin count and circuit board area for signal routing, the HI-6131 is the apparent winner, but do not make a hasty decision without considering the performance differences.

The HI-6130 bus interface reads or writes one word at a time, with a cycle time of 150 ns. Reads and writes are random access, and can occur in any order. To speed up a multiword read sequences from sequential addresses, the HI-6130 automatically pre-fetches the following RAM or register location after each single-word read cycle. When reading sequential addresses, the device asserts WAIT while fetching the first data location, then all sequential address (pre-fetched) reads that follow are faster, performed without asserting WAIT.

The HI-6131 data transfer speed depends on the SPI clock frequency provided by the MCU SPI interface. When the SPI is clocked at the maximum SCK frequency, 20 MHz, each word is transferred in 800 ns, plus the overhead associated with SPI op code execution. The maximum SCK frequency for many MCUs is 15 MHz, resulting in a 16-bit word transfer time of 1,067 ns, plus op code execution overhead and C program handling for the MCU SPI peripheral, which can be significant.

A HI-6131 Memory Address Pointer (register) is initialized by the MCU or FPGA before a read or write operation begins. The read/write operation is then initiated using an 8-bit SPI op code, serially shifted into the HI-6131 SPI by the MCU or FPGA. The host then continues clocking SCK in 16-clock multiples to read or write successive RAM or register addresses. As long as clocking continues, successive addresses are read or written. Potential problems occur when interrupts are enabled during a multi-word access. If the program's interrupt handler seizes the SPI bus to service the interrupt, it potentially disrupts an unfinished multi-word transfer. Without proper software design, a simple return-from-interrupt results in a broken multi-word transfer because the hardware doesn't know that an interrupt occurred and the Memory Address Pointer may or may not contain the RAM or register address for the next location in the interrupted multi-word transfer.

During HI-6131 SPI transfers, interrupts must be disabled. The simplest implementation disables interrupts before sending the op code, then re-enables interrupts after reading or writing the last word in the multi-word transfer. If this causes unacceptable interrupt latency, some careful software design is needed. With suitable precautions, interrupts can be momentarily re-enabled then immediately disabled between SPI words. A pending interrupt that occurred during the interrupt-off interval will be immediately recognized when interrupts are re-enabled. The pending interrupt's service routine will execute; the return-from-interrupt will jump to and execute the following "disable interrupt" statement.

The example HI-6131 software successfully completes an interrupted multi-word sequence by using a "SPI interrupt occurred" flag, tested each time interrupts are momentarily disabled then re-enabled between SPI words. Upon detection of interrupt servicing between words, the program re-initializes the memory address pointer for the next word, then re-issues the original op code to resume the interrupted multi-word transfer. This approach works, but requires diligence when compared to the carefree interrupt behavior of the HI-6130. Nesting interrupts 3 or more levels would be challenging.

The last issue when choosing between HI-6130 and HI-6131 is ease of debug when using IAR Embedded Workbench®. Because HI-6130 registers and RAM are memory-mapped, a debugger Memory window displays data for a range of addresses, which automatically updates each time program execution stops. Changed data values are displayed in red. The HI-6130 design example defines pointer-referenced C structures (structs) which simplify addressing of the various BC, RT and monitor RAM tables and other structures. The defined C structs allow use of debugger Watch windows for observing register or RAM table data, updated each time program execution stops.

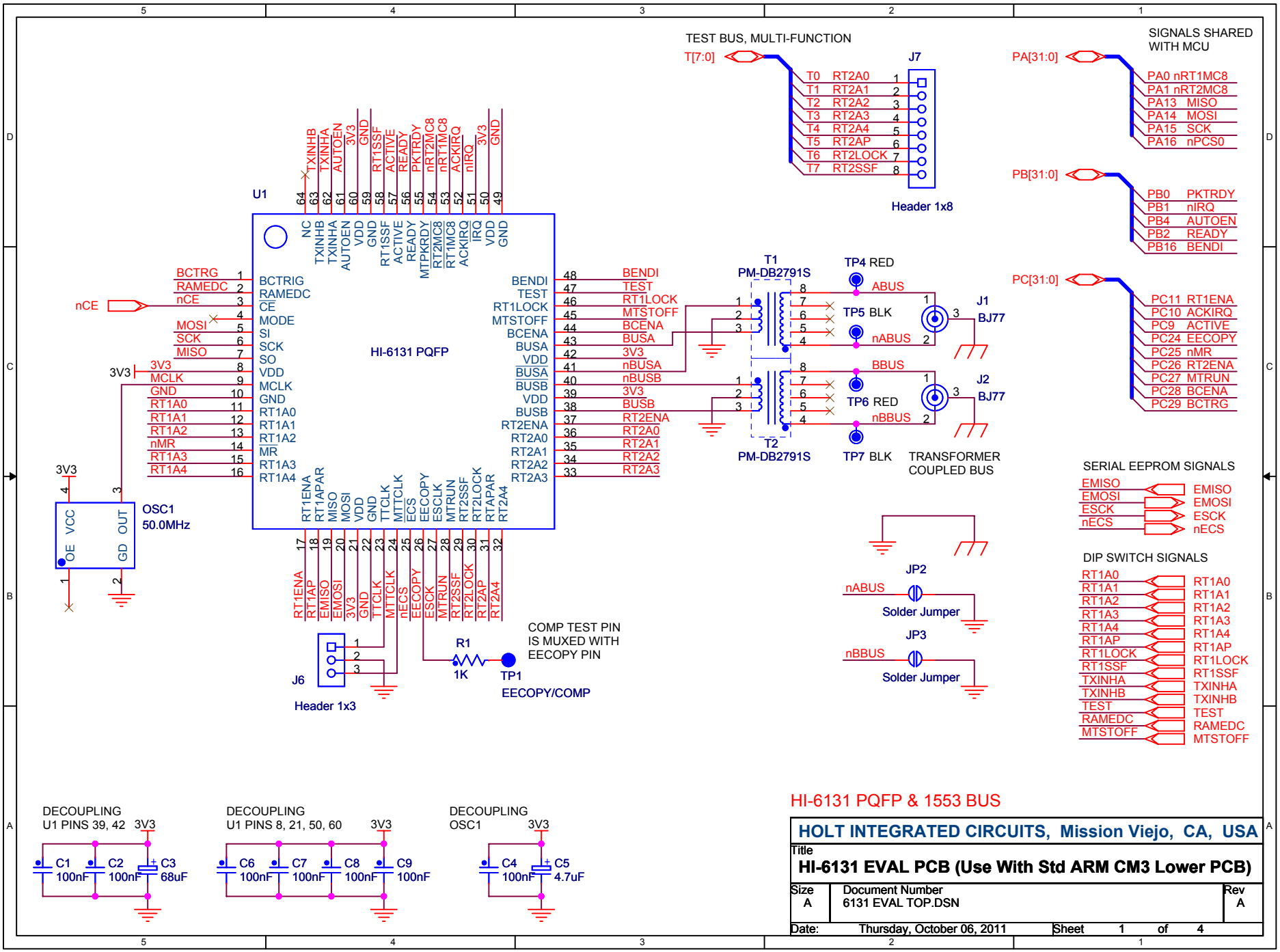
When using the HI-6131, SPI access is incompatible with C structures for table addressing. Register/RAM inspection tools comparable to the HI-6130 debugger Watch window are unavailable when using the HI-6131. Instead a utility function, like the HI-6131 demo program's Memory_watch() function, must be written in C to read a range of addresses, and format the data for display using console I/O or some other display means. Of course the application program must be running to call the display function when Memory_watch() is needed.

In summary…
With just 4 host interface signals for accessing RAM or registers, the HI-6131 SPI interface simplifies hardware design. This advantage is offset by significantly slower read/write access times, the need for carefully-designed interrupt handling software, and the lack of prepackaged register/RAM inspection tools when debugging.

In contrast, the HI-6130 bus interface provides higher-speed random access, programming ease and powerful debugger tools. These advantages are offset by the need to connect 36 interface signals for accessing RAM or registers. Compared to the HI-6131, two additional 16-bit wide buses (address and data) must be connected between the HI-6130 and its host MCU.

Bill of Materials **HI-6131 Evaluation Board, Top** **Revised: July 19, 2011**

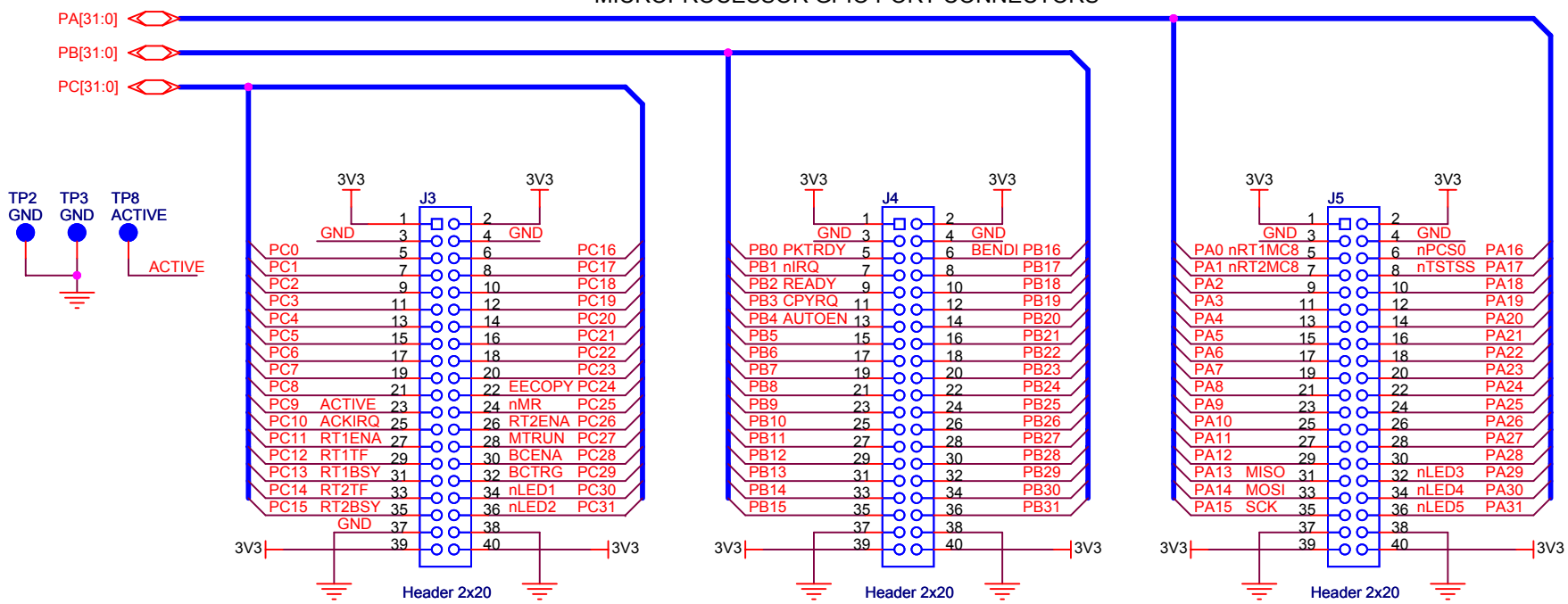
Item	*Qty*	*Description*	*Reference*	*DigiKey*	*Mfr P/N*
1	1	PCB, Bare, Eval Board	N/A	--------	--------
2	9	Capacitor, Ceramic 0.1uF 20% 50V Z5U 0805	C1,C2,C5, C6,C7,C8, C9,C10,C11	399-1176-1-ND	Kemet C0805C104M5UACTU
3	1	Capacitor, Ceramic 4.7uF 10% 6.3V X5R 0805	C5	399-3134-1-ND	Kemet C0805C475K9PACTU
4	1	Capacitor 68uF 10% 6.3V Tantalum 400 mOhm SMD EIA 6032-28	C3	495-1507-1-ND	Kemet B45197A1686K309
5	2	Connector 3-Lug Concentric Triax Bayonet Jack, Panel Front Mount TRB (BJ77)	J1,J2	MilesTek 10-06570	Trompeter Electronics BJ77 Use 0.469" Round Hole
6	1	Header, Male 2x10 0.1" Pitch, 0.230" Pins, 0.120" Tails	J3	S2012E-10-ND	Sullins
7	1	Header, Male 2x7 0.1" Pitch, 0.230" Pins, 0.120" Tails	J4	S2012E-07-ND	Sullins
8	1	Header, Male 2x4 0.1" Pitch, 0.230" Pins, 0.120" Tails	J5A	S2012E-04-ND	Sullins
9	1	Header, Male 2x5 0.1" Pitch, 0.230" Pins, 0.120" Tails	J5B	S2012E-05-ND	Sullins
10	------	Header, 1x3, 0.1" pitch	J6	DO NOT STUFF	--------
11	------	Header, 1x8, 0.1" pitch	J7	DO NOT STUFF	--------
12	1	Header, 5x3, 0.1" pitch	JP1	Samtec	Samtec TSW-105-07-T-T
13	5	Jumper, shorting, w/ grip, 0.1"	JP1	S9341-ND	Sullins NPC02SXON-RC
14		Solder Jumper	JP2,JP3	DO NOT SOLDER	--------
15	1	LED Yellow 0805	LED5	160-1175-1-ND	Lite On LTST-C170YKT
16	3	LED Green 0805	LED1 - LED3	160-1179-1-ND	LiteOn LTST-C170GKT
17	1	LED Red 0805	LED4	160-1176-1-ND	LiteOn LTST-C170CKT
18	1	Osc, 50.00MHz 25ppm 3.3V SMD 5mm x 7mm	OSC1	CTX328LVCT-ND	CTX CB3LV-3I-64M0000-T
19	5	Resistor, 150 5% 1/8W 0805	R8,R9,R10, R11,R12	P150ACT-ND	Any
20	1	Resistor, 1.0K 5% 1/8W 0805	R1	P1.0KACT-ND	Any
21	1	Resistor, 2.2K 5% 1/8W 0805	R17	P2.2KACT-ND	Any
22	2	Resistor, 10K 5% 1/8W 0805	R30,R31	P10KACT-ND	Any
23	6	Resistor, 47K 5% 1/8W 0805	R13,R14,R15 R16,R18,R19	P47KACT-ND	Any
24	2	DIP Switch 5-Position SMD	SW1,SW4	CT2195LPST-ND	CTS 219-5LPST
25	2	DIP Switch 6-Position SMD	SW5,SW6	CT2196LPST-ND	CTS 219-6LPST
26	1	DIP Switch 10-Position SMD	SW8	CT21910LPST-ND	CTS 219-10LPST
27	1	Slide Switch SPDT SMD	SW2	563-1022-1-ND	Copal CJS-1200TB
28	2	Transformer MIL-STD-1553 Single, 1:2.50, PM-DB2791S	T1,T2	Holt PM-DB2791S	Premier Magnetics PM-DB2791S
29	------	Test Point, pad w/ plated hole	TP1,TP9	------	--------
30	2	Test Point, Red Insulator, 0.062" hole	TP4,TP6	5010K-ND	Keystone 5010
31	3	Test Point, Black Insulator, 0.062" hole	TP2,TP3,TP5, TP7	5011K-KD	Keystone 5011
32	1	Test Point, White Insulator, 0.062" hole	TP8	5012K-KD	Keystone 5012
33	1	IC HI-6131 Holt 64-PQFP	U1	------	--------
34	2	IC, Serial EEPROM 512Kbit 20MHz SPI 8-SOIC, Microchip	U2, U3	25LC512T-I/SNCT-ND	Microchip 25LC512T-I/SN

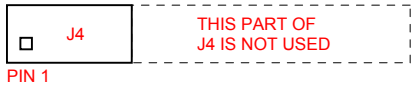


HI-6131 PQFP & 1553 BUS

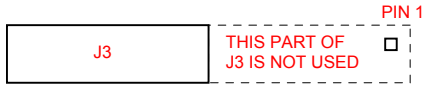
HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: HI-6131 EVAL PCB (Use With Std ARM CM3 Lower PCB)		
Size: A	Document Number: 6131 EVAL TOP.DSN	Rev: A
Date: Thursday, October 06, 2011	Sheet 1 of 4	

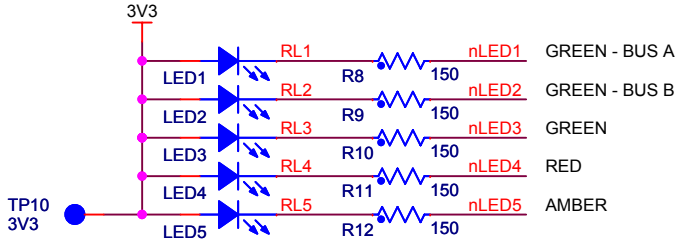
MICROPROCESSOR GPIO PORT CONNECTORS





HEADER ORIENTATION
ON THE CIRCUIT BOARD

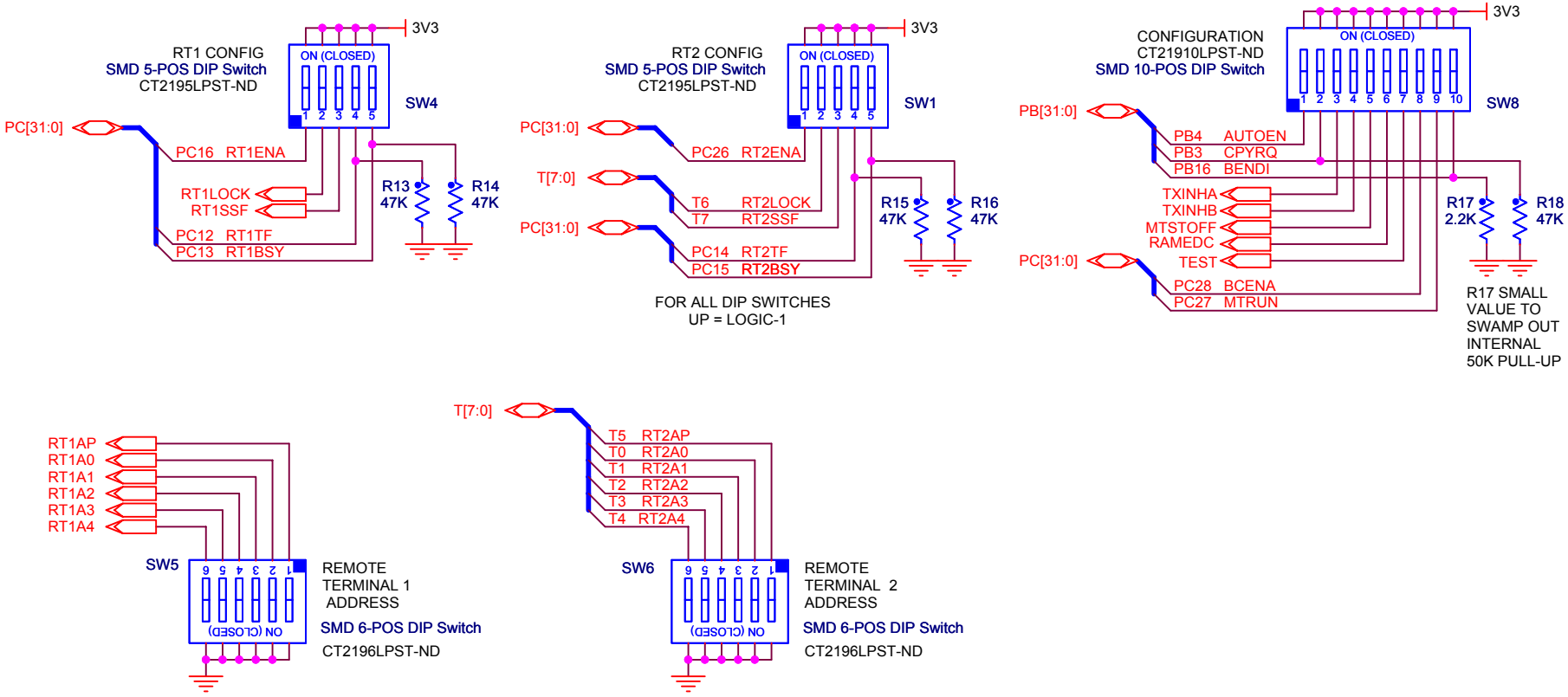




STATUS LEDS

MCU I/O HEADERS & LEDS

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: HI-6131 EVAL PCB (Use With Std ARM CM3 Lower PCB)		
Size: A	Document Number: 6131 EVAL TOP.DSN	Rev: A
Date: Thursday, October 06, 2011	Sheet 2 of 4	



ALL DIP SWITCH SIGNALS WITH A PULL-UP OR A PULL-DOWN GO TO THE MCU.

DIP SWITCHES & SERIAL EEPROM

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA

Title: **HI-6131 EVAL PCB (Use With Std ARM CM3 Lower PCB)**

Size	Document Number	Rev
A	6131 EVAL TOP.DSN	A

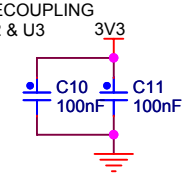
Date: Thursday, October 06, 2011 Sheet 3 of 4

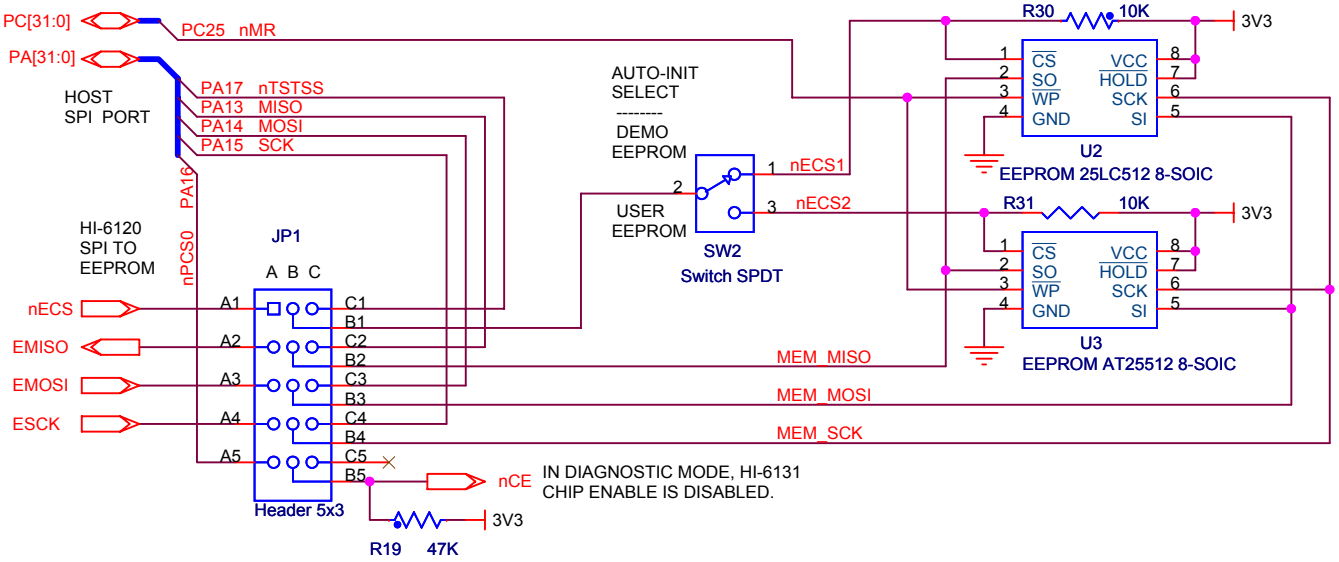
DUAL EEPROM CIRCUIT FOR EVALUATION BOARD ONLY. SW2 SELECTS EEPROM.
JP1 JUMPER ALSO PROVIDES MCU READ/WRITE ACCESS TO SELECTED EEPROM.

JUMPERS NORMALLY SPAN COLUMNS A-B FOR HI-613X CONTROL OF SERIAL EEPROM.

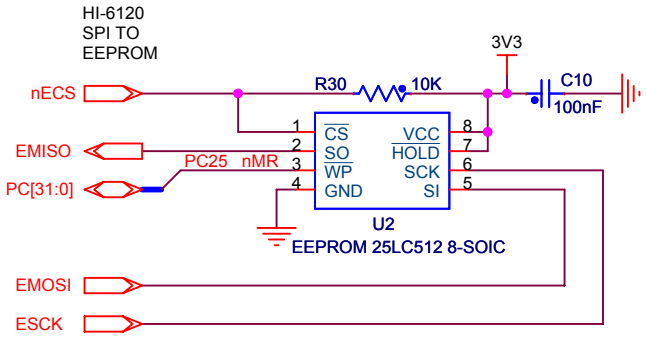
FOR DIAGNOSTIC TESTS, JUMPERS SPAN COLUMNS B-C SO THE MCU SPI CAN READ/WRITE THE SERIAL EEPROM. THE C PROGRAM CONTROLS TEST SLAVE SELECT SIGNAL, nTSTSS.

DECOUPLING U2 & U3





TYPICAL APPLICATION REPLACES ABOVE EEPROM CIRCUIT WITH THIS SIMPLE CONFIGURATION



HI-6131 PQFP & 1553 BUS

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA

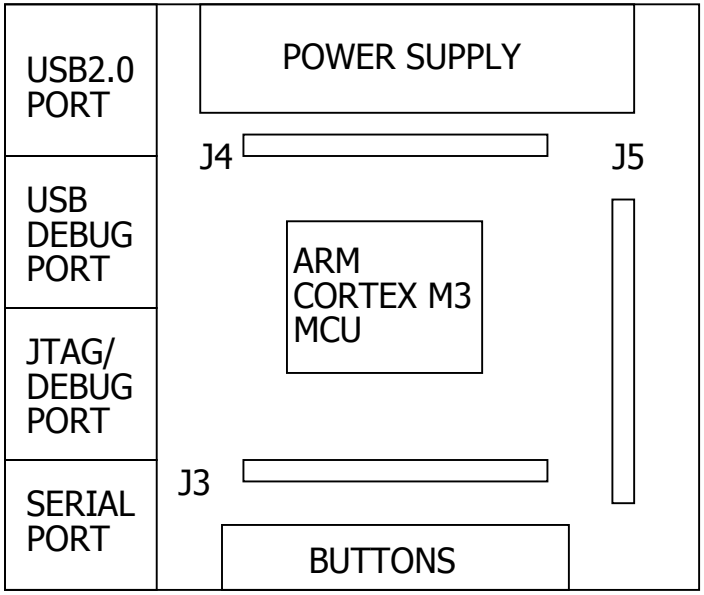
Title: **HI-6131 EVAL PCB (Use With Std ARM CM3 Lower PCB)**

Size	Document Number	Rev
A	6131 EVAL TOP.DSN	A

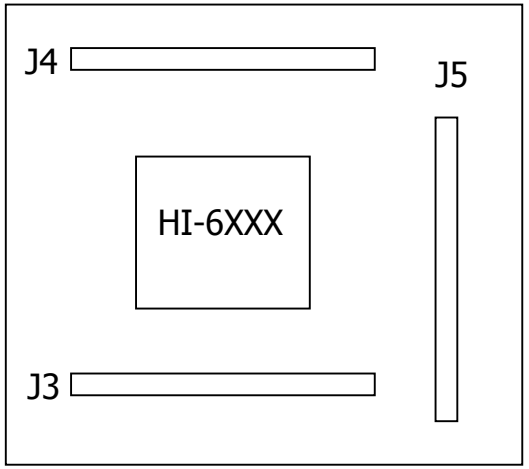
Date: Thursday, October 06, 2011 Sheet 4 of 4

Bill of Materials
ARM Cortex M3 MCU Board
Rev. E

Item	Qty	Description	Reference	DigiKey	Mfr P/N
1					
2	1	PCB, Bare, Evaluation Board	N/A	--------	
3	1	Ferrite Bead, 220 Ohm @ 100MHz 300mA DC 0805	FB1	732-1602-1-ND	Wurth 742792034
4	2	Capacitor, Ceramic 10nF 10% 50V X7R 0603	C1,C42	490-1512-1-ND	Murata GRM188R71H103KA01D
5	2	Capacitor, Ceramic 10pF 10% NP0 C0G 0V 0603	C23,C34	490-1403-1-ND	Murata GRM1885C1H100JA01D
6	4	Capacitor, Ceramic 20pF 5% NP0 C0G 0V 0603	C14,C21,C25, C27	490-1410-1-ND	Murata GRM1885C1H200JA01D
7	29	Capacitor, Ceramic 100nF 10% 25V Y5V 0603	C2,C4,C6-C11, C13,C15-C19,C22,C24,C26,C28,C29,C33, C35-C40,C45-46,C54	490-1575-1-ND	Murata GRM188F51E104ZA01D
8	4	Capacitor, Tantalum 4.7uF 10% 10V Low ESR SMD 1206	C5,C20,C31, C32	478-2391-1-ND	AVX TPSA475K010R1400
9	4	Capacitor, Tantalum 10uF 10% 10V Low ESR SMD 1206	C3,C12,C30,C41	478-3317-1-ND	AVX TPSA106K010R1800
10	1	Capacitor 22uF 10% 6.3V Tantalum Low ESR SMD C	C43	399-10521-1-ND	Kemet T495C226K006ATE380
11	1	Capacitor 100uF 10% 6.3V Tantalum Low ESR SMD C	C44	495-1509-1-ND	Kemet T495C107K006ZTE150
12	1	Header, Male Shrouded 2x10, 0.1" Pitch	J1	HRP20H-ND	Assmann AWHW20G-0202-T
13	1	Connector, Receptacle USB Mini B Rt-Angle PCB Mount	J2	H2959CT-ND	Hirose UX60-MB-5ST
14	1	Connector DB9F, Right-Angle PCB Short Body, Board Lock	J6	AE10924-ND	Assman A-DF-09-A/KG-T4S
15	1	Jack, DC Power, 2.5mm ID x 2.1mm pin	J7	CP-102AH-ND	Cui PJ-102AH
16	3	Receptacle, Female 2x20, 0.1" Pitch, 8.5mm Height, 3.2mm Solder Tails	J3,J4,J5	S6104-ND	Sullins PPTC202LFBN-RC
17	1	Solder Jumper	JP1	SOLDER OPEN	
18	2	Inductor, 10uH,100mA 0805	L1,L2	490-4029-1-ND	Murata LQM21FN100M70L
19	1	LED Green 0805	LED1	160-1179-1-ND	LiteOn LTST-C170GKT
20	0	Resistor, Prov 1/8W 0805	R1,R15,R16, R44,R45	DO NOT STUFF	
21	7	Resistor, 0 ohm 1/8W 0805	R9,R12,R13, R14,R22,R23, R29	P0.0ACT-ND	Panasonic ERJ-6GEY0R00V
22	2	Resistor, 1.0 5% 1/8W 0805	R7,R8	P1.0ACT-ND	Panasonic ERJ-6GEYJ1R0V
23	2	Resistor, 39 5% 1/8W 0805	R4,R5	P39ACT-ND	Panasonic ERJ-6GEYJ390V
24	1	Resistor, 150 5% 1/8W 0805	R17	P150ACT-ND	Panasonic ERJ-6GEYJ151V
25	1	Resistor, 4.7K 5% 1/8W 0805	R3	P4.7KACT-ND	Panasonic ERJ-6GEYJ472V
26	1	Resistor, 6.8K 5% 1/8W 0805	R6	P6.8KACT-ND	Panasonic ERJ-6GEYJ682V
27	0	Resistor, 47K 5% 1/8W 0805	R18	DO NOT STUFF	Panasonic ERJ-6GEYJ473V
28	0	Resistor, 68K 5% 1/8W 0805	R19	DO NOT STUFF	Panasonic ERJ-6GEYJ683V
29	11	Resistor,100K 5% 1/8W 0805	R2,R10,R11, R20,R21,R24, R25,R26,R27, R28,R42	P100KACT-ND	Panasonic ERJ-6GEYJ104V
30	3	Switch Tactile SPST 6 x 6 mm SMT	SW1,SW2,SW3	P12932SCT-ND	Panasonic EVQ-Q2B03W
31	2	Test Point, Black Insulator, 0.062" hole	TP2,TP3	5011K-ND	Keystone 5011
32	1	Test Point, Red Insulator, 0.062" hole	TP1	5010K-ND	Keystone 5010
33	1	IC, MCU 32-Bit 256KB Flash, 144-LQFP	U1	ATSAM3U4EA-AU-ND	Atmel ATSAM3U4EA-AU
34	1	4-Ch TVS ESD Protection SOT23-6	U2	296-28203-1-ND	TI TPD4E001DBVR
35	1	IC, RS232 Driver/Receiver 3.0 to 5.5VDC 16-SOIC (3.9mm wide)	U3	296-19752-1-ND	Texas Inst MAX3232EIDR
36	1	IC Voltage Regulator 3.3V 1A LDO, SOT-223	U5	497-1228-1-ND	ST Micro LD1117AS33TR
37	1	PolyZen 5.6V PPTC protected Zener SMD	U6	ZEN056V130A24LSCT-ND	TE ZEN056V130A24LS
38	1	Filter, EMI 35dB 10A 1MHz-1GHz SMD	U7	490-5052-1-ND	Murata BNX022-01L
39	1	IC Voltage Ref 2.5V 1% Micropower SOT-23	VR1	576-1047-1-ND	Micrel LM4040DYM3-2.5
40	1	Crystal 12.00MHz, 50ppm 20pF, HC-49US leaded	Y1	631-1105-ND	Fox FOXSLF/120-20
41	1	Crystal, 32768 Hz 12.5pF cylinder leaded	Y2	535-9033-1-ND	Abracon AB26TRB-32.768KHZ-T
42	5	Rubber Foot, Bumpon Black Hemisphere, .312 X.200 H	Place at 4 corners and center	SJ5746-0-ND	3M SJ61A1
47	1	Capacitor, Ceramic 100nF, -20% / +80% 25V Y5V 0603	C66	490-1575-1-ND	Murata GRM188F51E104ZA01D
48	1	Capacitor, Ceramic 33pF, 5% 50V C0G 0603	C59	490-1415-1-ND	Murata GRM1885C1H330JA01D
49	2	Capacitor, Ceramic 15pF, 5% 50V C0G 0603	C60,C61	490-1407-1-ND	Murata GRM1885C1H150JA01D
54	1	Ferrite Bead, 220 Ohm @ 100MHz 300mA DC 0805	FB2	732-1602-1-ND	Wurth 742792034
55	1	Solder Jumper	JP2	SOLDER OPEN	
56	1	Connector, Receptacle USB Mini B Rt-Angle PCB Mount	J8	H2959CT-ND	Hirose UX60-MB-5ST
57	1	LED Green 0805	LED2	160-1179-1-ND	LiteOn LTST-C170GKT
59	1	Resistor, 220 ohm 5% 1/10W 0603	R31	P220GCT-ND	Panasonic ERJ-3GEYJ221V
63	2	Resistor, 27 ohm 5% 1/10W 0603	R36,R38	P27GCT-ND	Panasonic ERJ-3GEYJ270V
66	1	4-Ch TVS ESD Protection SOT23-6	U4	296-28203-1-ND	TI TPD4E001DBVR



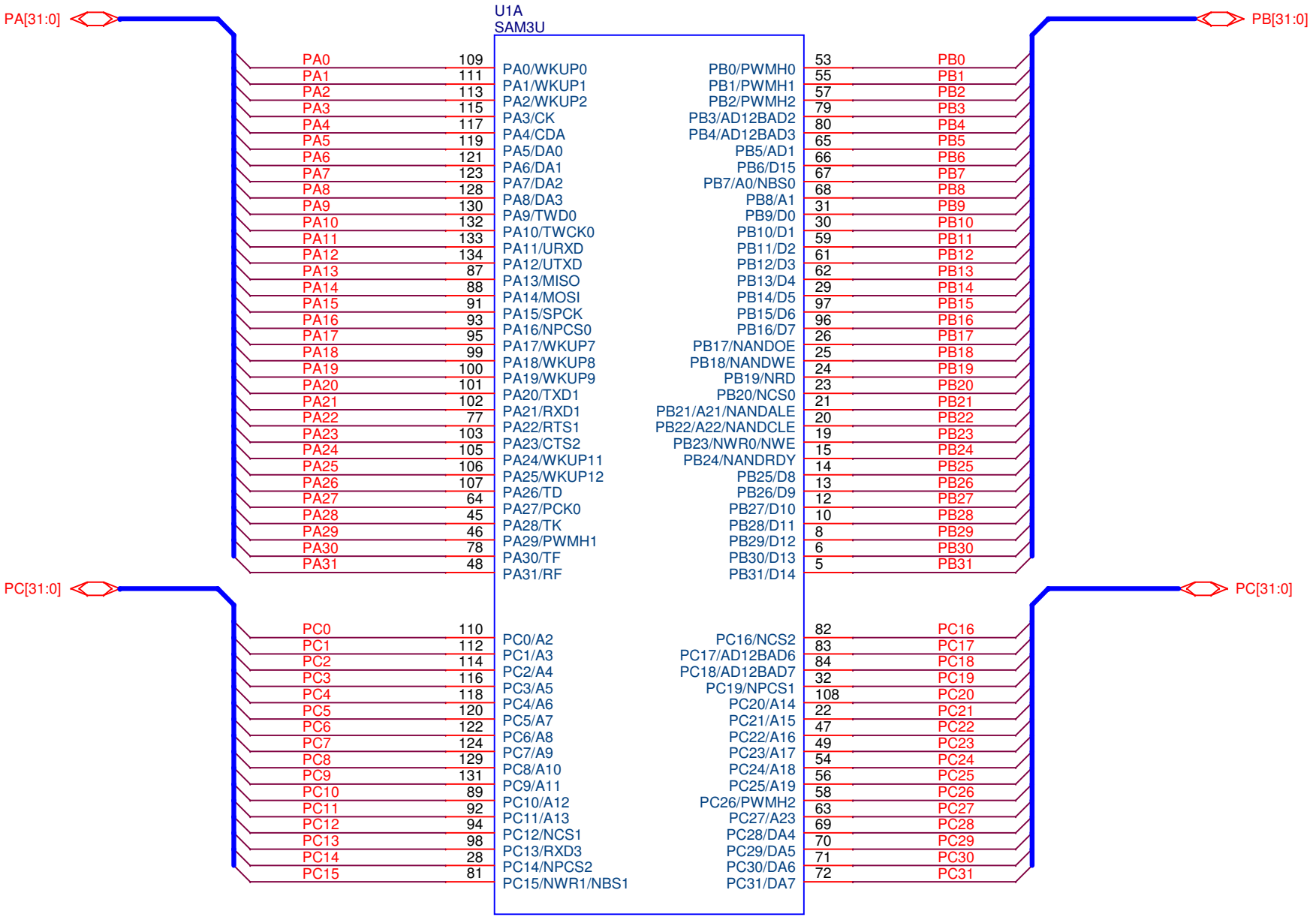
LOWER CIRCUIT BOARD



STACKING UPPER CIRCUIT BOARD

J3,J4 & J5 ARE DUAL-ROW STACKING RECEPTACLES (LOWER BOARD) AND HEADERS (UPPER BOARD).

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: ARM CORTEX M3 MICROCONTROLLER BOARD		
Size: A	Document Number: CM3 BOARD REV E.DSN	Rev: E
Date: Wednesday, June 01, 2016	Sheet 1 of 7	



ARM CORTEX M3 PIO

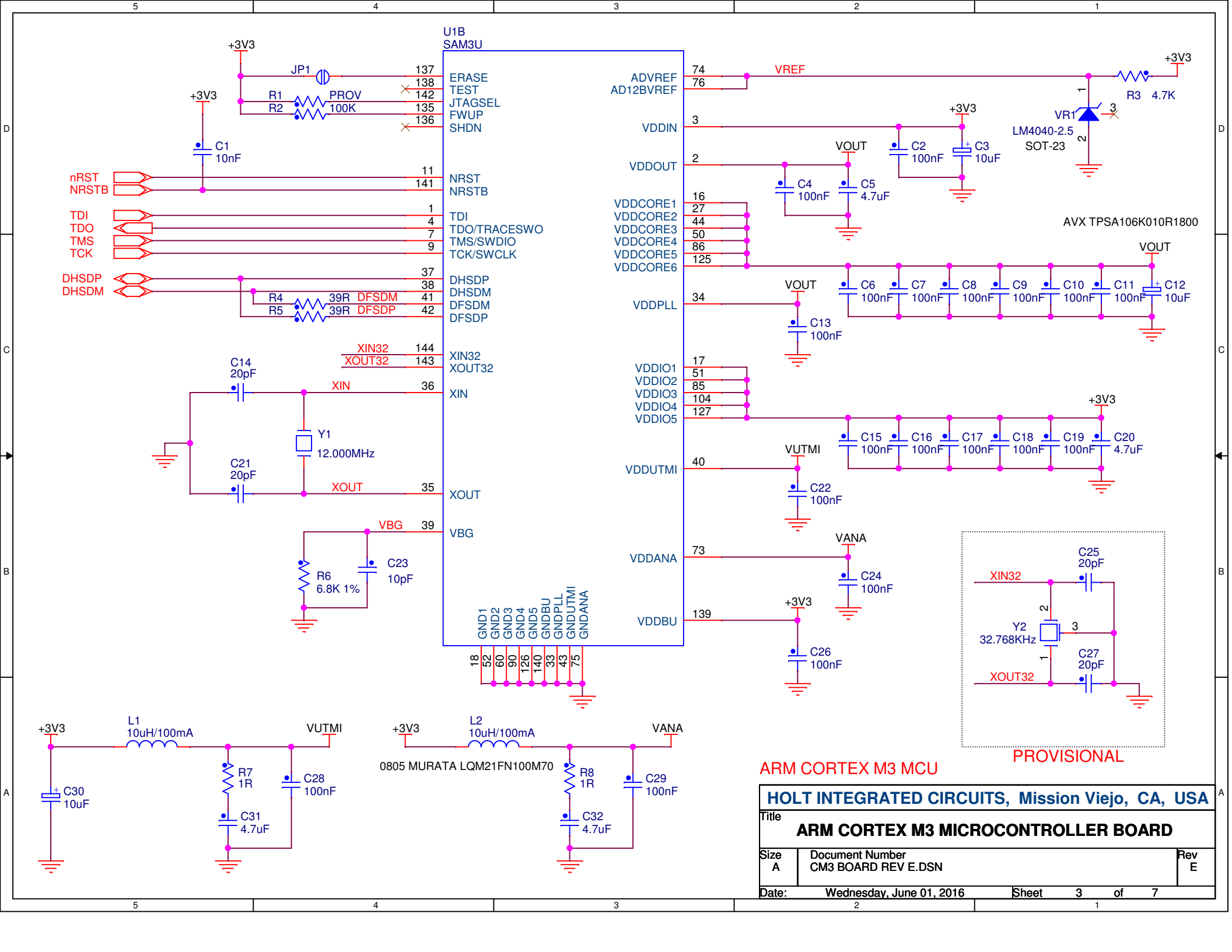
HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA

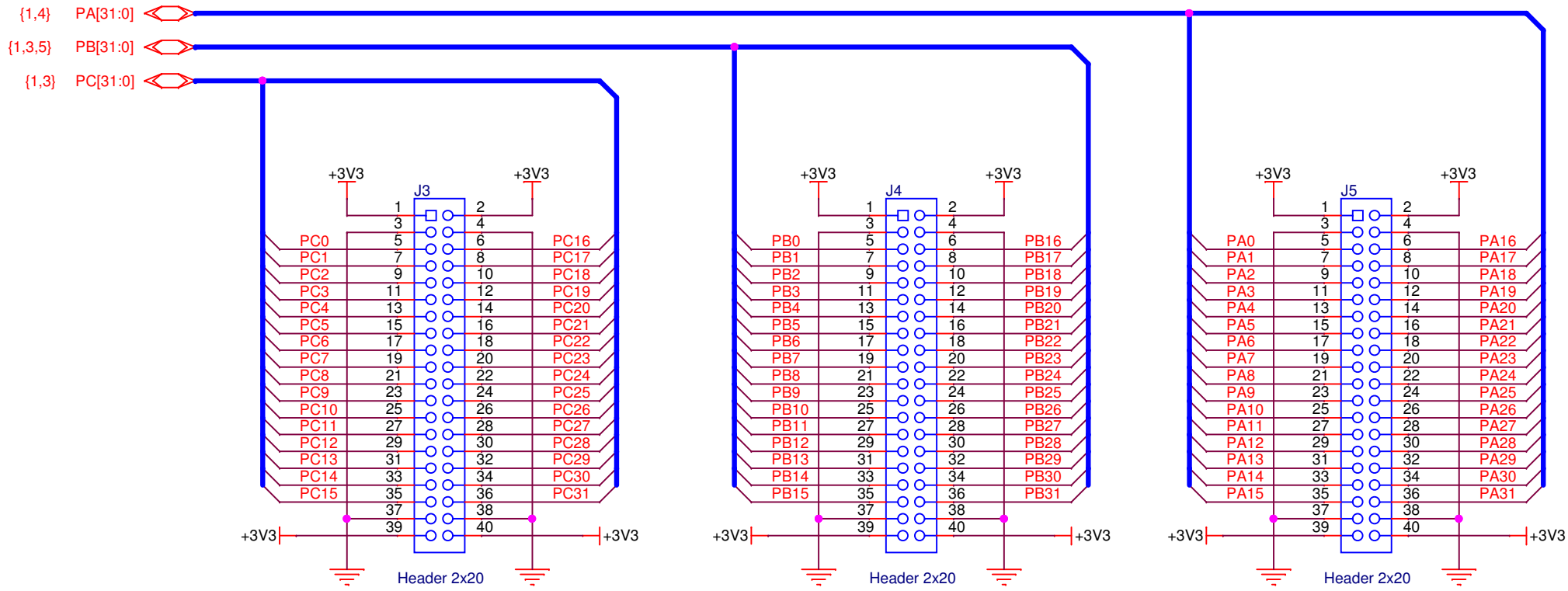
Title

ARM CORTEX M3 MICROCONTROLLER BOARD

Size	Document Number	Rev
A	CM3 BOARD REV E.DSN	E

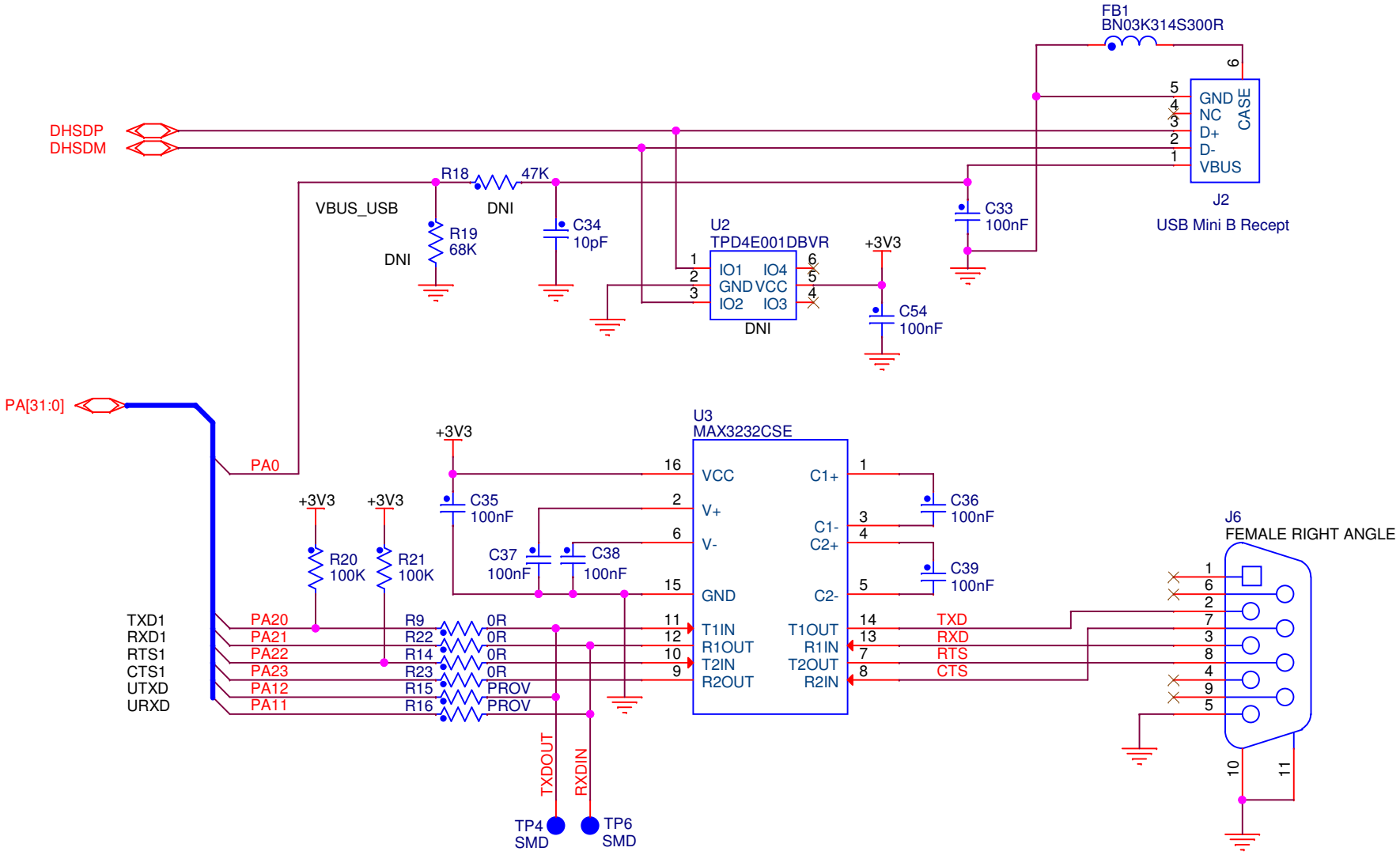
Date: Wednesday, June 01, 2016 Sheet 2 of 7


U1B
SAM3U
+3V3
JP1
R1
R2
PROV
100K
C1
10nF
nRST
NRSTB
TDI
TDO
TMS
TCK
DHSDP
DHSDM
R4
R5
39R
DFSDM
DFSDP
XIN32
XOUT32
C14
20pF
XIN
Y1
12.000MHz
C21
20pF
XOUT
VBG
R6
6.8K 1%
C23
10pF
ERASE
TEST
JTAGSEL
FWUP
SHDN
NRST
NRSTB
TDI
TDO/TRACESWO
TMS/SWDIO
TCK/SWCLK
DHSDP
DHSDM
DFSDM
DFSDP
XIN32
XOUT32
XIN
XOUT
VBG
GND1
GND2
GND3
GND4
GND5
GNDBU
GNDPLL
GNDUTMI
GNDANA
ADVREF
AD12BVREF
VDDIN
VDDOUT
VDDCORE1
VDDCORE2
VDDCORE3
VDDCORE4
VDDCORE5
VDDCORE6
VDDPLL
VDDIO1
VDDIO2
VDDIO3
VDDIO4
VDDIO5
VDDUTMI
VDDANA
VDDBU
VREF
R3 4.7K
VR1
LM4040-2.5
SOT-23
VOUT
C2
100nF
C3
10uF
C4
100nF
C5
4.7uF
AVX TPSA106K010R1800
C6
100nF
C7
100nF
C8
100nF
C9
100nF
C10
100nF
C11
100nF
C12
10uF
C13
100nF
C15
100nF
C16
100nF
C17
100nF
C18
100nF
C19
100nF
C20
4.7uF
VUTMI
C22
100nF
VANA
C24
100nF
C26
100nF
C25
20pF
Y2
32.768KHz
C27
20pF
L1
10uH/100mA
C30
10uF
R7
1R
C28
100nF
C31
4.7uF
L2
10uH/100mA
0805 MURATA LQM21FN100M70
R8
1R
C29
100nF
C32
4.7uF
PROVISIONAL
ARM CORTEX M3 MCU
HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA
Title
ARM CORTEX M3 MICROCONTROLLER BOARD
Size
A
Document Number
CM3 BOARD REV E.DSN
Rev
E
Date:
Wednesday, June 01, 2016
Sheet
3
of
7



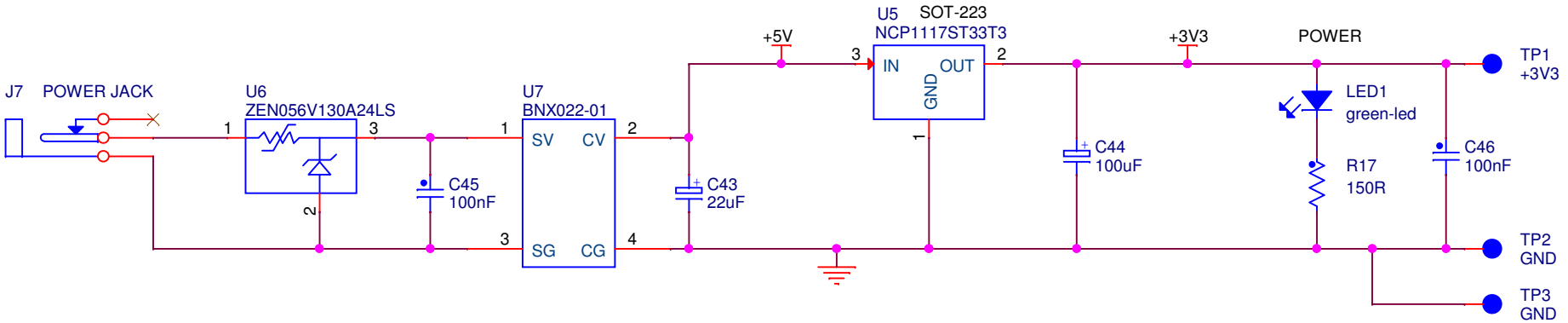
BOARD I/O HEADERS, BUTTONS

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: ARM CORTEX M3 MICROCONTROLLER BOARD		
Size: A	Document Number: CM3 BOARD REV E.DSN	Rev: E
Date: Wednesday, June 01, 2016	Sheet 4 of 7	



USB & RS-232 SERIAL

HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: ARM CORTEX M3 MICROCONTROLLER BOARD		
Size: A	Document Number: CM3 BOARD REV E.DSN	Rev: E
Date: Wednesday, June 01, 2016	Sheet 5 of 7	



POWER SUPPLY

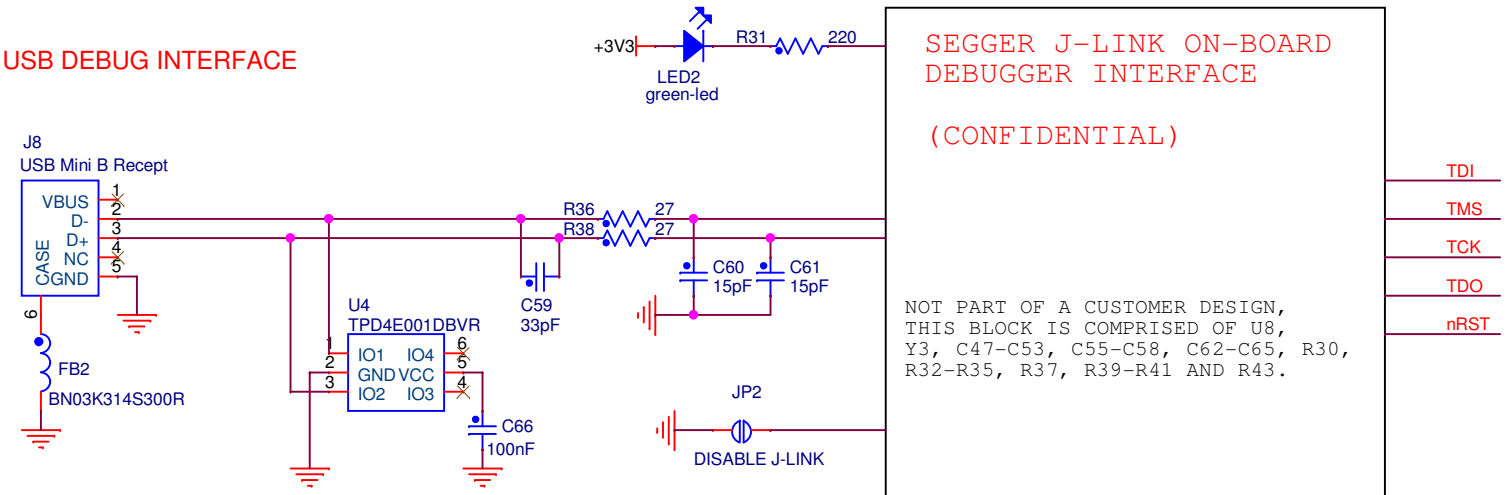
HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA

Title
ARM CORTEX M3 MICROCONTROLLER BOARD

Size	Document Number	Rev
A	CM3 BOARD REV E.DSN	E

Date: Wednesday, June 01, 2016 Sheet 6 of 7

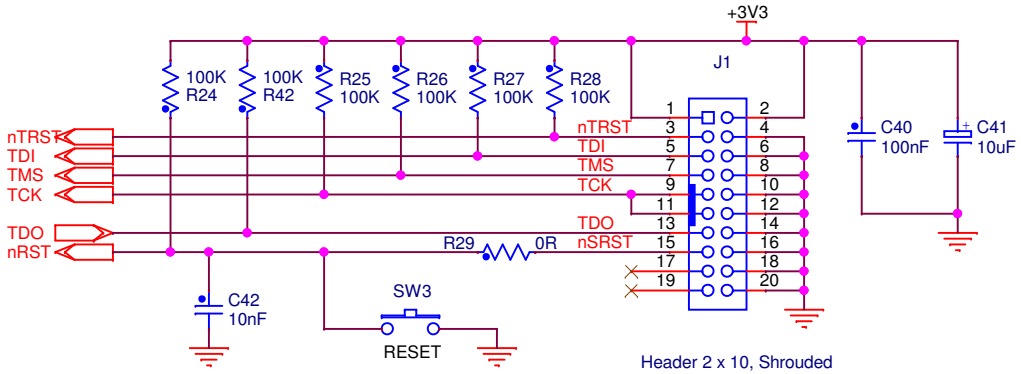
USB DEBUG INTERFACE



DEBUGGER INTERFACE COPIED FROM ATMEL ARM CORTEX M3

USE THIS TO CONNECT J-LINK IF ABOVE CIRCUITRY IS NOT POPULATED OR WHEN IT IS DISABLED BY JUMPER JP2.

PARALLEL DEBUG INTERFACE



HOLT INTEGRATED CIRCUITS, Mission Viejo, CA, USA		
Title: ARM CORTEX M3 MICROCONTROLLER BOARD		
Size: Custom	Document Number: CM3 BOARD REV E.DSN	Rev: E
Date: Wednesday, June 01, 2016	Sheet 7 of 7	

REMOTE TERMINAL RT1 MEMORY MAP FOR HI-6130 AND HI-6131 APPLICATION DEVELOPMENT BOARD PROGRAM

	dec	hex
Descriptor Table Base Addr	1024	0400
First Buffer Address	2048	0800

Descriptor Table Sector	Device Internal Addr same as HI-6131 Addr		Data Bus Addr Hex HI-6130 Only	
	Start	*End*	*Start*	*End*
Receive Subaddresses	0400	047F	60000800	600008FE
Transmit Subaddresses	0480	04FF	60000900	600009FE
Receive Mode Codes	0500	057F	60000A00	60000AFE
Transmit Mode Codes	0580	05FF	60000B00	60000BFE

Buffer Assignments for Receive and Transmit Subaddresses

Receive (Rx) Subaddress or Transmit (Tx) Subaddress	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Rx SA1	ping-pong	DPA	34	0800	0821	60001000	60001042	MIW + TT + 32 words
(data pointers A, B and		DPB	34	0822	0843	60001044	60001086	same
broadcast data pointer)		BDP	34	0844	0865	60001088	600010CA	same
Tx SA1	ping-pong	DPA	34	0866	0887	600010CC	6000110E	same
(data pointers A, B and		DPB	34	0888	08A9	60001110	60001152	same
broadcast data pointer)		BDP	4	08AA	08AD	60001154	6000115A	MIW + TT + 2 pad
Rx SA30 and Tx SA30 for data wrap-around	index-0	DPA	34	08AE	08CF	6000115C	6000119E	MIW + TT + 32 words
Rx SA2	index-32	DPA	1088	08D0	0D0F	600011A0	60001A1E	32 x (MIW + TT + 32 words)
		BDP	34	0D10	0D31	60001A20	60001A62	
Tx SA2	index-32	DPA	1088	0D32	1171	60001A64	600022E2	
		BDP	4	1172	1175	600022E4	600022EA	MIW + TT + 2 pad
Rx SA3	circ1-32	DPA	1088	1176	15B5	600022EC	60002B6A	32 x (MIW + TT + 32 words)
		pad	32	15B6	15D5	60002B6C	60002BAA	pad for overrun
Tx SA3	circ1-32	DPA	1088	15D6	1A15	60002BAC	6000342A	32 x (MIW + TT + 32 words)
		pad	32	1A16	1A35	6000342C	6000346A	pad for overrun
shared buffer: all unimplemented Rx subaddresses	index-0	DPA	34	1A36	1A57	6000346C	600034AE	MIW + TT + 32 words
shared buffer: all unimplemented Tx subaddresses	index-0	DPA	34	1A58	1A79	600034B0	600034F2	MIW + TT + 32 words
RAM assigned below (MCs)	---	---	142	1A7A	1B07	600034F4	6000360E	
unassigned RAM	---	---	72	1B08	1B4F	60003610	6000369E	
assigned to BC	---	---	176	1B50	1BFF	600036A0	600037FE	BC Mode Command Data BC Instruction List
Rx & Tx SA4	circ-2 256 msg max	MIB	512	1C00	1DFF	600036A0	60003BFE	256 x (MIW + TT)
		DPA	8192	1E00	3DFF	60003C00	60007BFE	
assigned to BC	---	---	256	3E00	3EFF	60007C00	60007DFE	BC Msg Control Blocks
unassigned RAM	---	---	256	3F00	3FFF	60007C00	60007FFE	

Shared Buffer Assignments for Undefined and Reserved Mode Code Commands

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Undefined & Reserved Receive (Rx) Mode Codes Transmit (Tx) Mode Codes	Buffer Method and Data Pointer(s)	Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
shared buffer: undefined Rx MC0 - MC15	index-0 DPA	4	1A7A	1A7D	600034F4	600034FA	MIW + TT, 0 data, 2 pad
shared buffer: undefined Rx MC16, undefined Rx MC18 - MC19, reserved Rx MC22 - MC31	index-0 DPA	4	1A7E	1A81	600034FC	60003502	MIW + TT, 1 data, 1 pad
shared buffer: undefined Tx MC9 - MC15	index-0 DPA	4	1A82	1A85	00803504	6000350A	MIW + TT, 0 data, 2 pad
shared buffer: undefined Tx MC17, undefined Tx MC20 - MC21, reserved Tx MC22 - MC31	index-0 DPA	4	1A86	1A89	6000350C	60003512	MIW + TT, 1 data, 1 pad

Buffer Assignments for Defined Transmit Mode Code Commands MC0 - MC8 (No Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Transmit (Tx) Mode Code Commands, No Data	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Tx MC0	ping-pong	DPA	2	1A8A	1A8B	60003514	60003516	MIW + TT
		DPB	2	1A8C	1A8D	60003518	6000351A	same
		BDP	2	1A8E	1A8F	6000351C	6000351E	same
Tx MC1	ping-pong	DPA	2	1A90	1A91	60003520	60003522	MIW + TT
		DPB	2	1A92	1A93	60003524	60003526	same
		BDP	2	1A94	1A95	60003528	6000352A	same
Tx MC2	ping-pong	DPA	2	1A96	1A97	6000352C	6000352E	MIW + TT
		DPB	2	1A98	1A99	60003530	60003532	same
		BDP	2	1A9A	1A9B	60003534	60003536	same
Tx MC3	ping-pong	DPA	2	1A9C	1A9D	60003538	6000353A	MIW + TT
		DPB	2	1A9E	1A9F	6000353C	6000353E	same
		BDP	2	1AA0	1AA1	60003540	60003542	same
Tx MC4	ping-pong	DPA	2	1AA2	1AA3	60003544	60003546	MIW + TT
		DPB	2	1AA4	1AA5	60003548	6000354A	same
		BDP	2	1AA6	1AA7	6000354C	6000354E	same
Tx MC5	ping-pong	DPA	2	1AA8	1AA9	60003550	60003552	MIW + TT
		DPB	2	1AAA	1AAB	60003554	60003556	same
		BDP	2	1AAC	1AAD	60003558	6000355A	same
Tx MC6	ping-pong	DPA	2	1AAE	1AAF	6000355C	6000355E	MIW + TT
		DPB	2	1AB0	1AB1	60003560	60003562	same
		BDP	2	1AB2	1AB3	60003564	60003566	same
Tx MC7	ping-pong	DPA	2	1AB4	1AB5	60003568	6000356A	MIW + TT
		DPB	2	1AB6	1AB7	6000356C	6000356E	same
		BDP	2	1AB8	1AB9	60003570	60003572	same
Tx MC8	ping-pong	DPA	2	1ABA	1ABB	60003574	60003576	MIW + TT
		DPB	2	1ABC	1ABD	60003578	6000357A	same
		BDP	2	1ABE	1ABF	6000357C	6000357E	same

Buffer Assignments for Defined Transmit Mode Code Commands MC16, MC18 and MC19 (1 Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Transmit (Tx) Mode Code Commands with Data Word	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Tx MC16	ping-pong	DPA	4	1AC0	1AC3	60003580	60003586	MIW + TT, 1 data, 1 pad
		DPB	4	1AC4	1AC7	60003588	6000358E	same
		BDP	4	1AC8	1ACB	60003590	60003596	same
Tx MC18	ping-pong	DPA	4	1ACC	1ACF	60003598	6000359E	MIW + TT, 1 data, 1 pad
		DPB	4	1AD0	1AD3	600035A0	600035A6	same
		BDP	4	1AD4	1AD7	600035A8	600035AE	same
Tx MC19	ping-pong	DPA	4	1AD8	1ADB	600035B0	600035B6	MIW + TT, 1 data, 1 pad
		DPB	4	1ADC	1ADF	600035B8	600035BE	same
		BDP	4	1AE0	1AE3	600035C0	600035C6	same

Buffer Assignments for Defined Receive Mode Code Commands MC17, MC20 and MC21 (1 Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Receive (Rx) Mode Code Commands with Data Word	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Rx MC17	ping-pong	DPA	4	1AE4	1AE7	600035C8	600035CE	MIW + TT, 1 data, 1 pad
		DPB	4	1AE8	1AEB	600035D0	600035D6	same
		BDP	4	1AEC	1AEF	600035D8	600035DE	same
Rx MC20	ping-pong	DPA	4	1AF0	1AF3	600035E0	600035E6	MIW + TT, 1 data, 1 pad
		DPB	4	1AF4	1AF7	600035E8	600035EE	same
		BDP	4	1AF8	1AFB	600035F0	600035F6	same
Rx MC21	ping-pong	DPA	4	1AFC	1AFF	600035F8	600035FE	MIW + TT, 1 data, 1 pad
		DPB	4	1B00	1B03	60003600	60003606	same
		BDP	4	1B04	1B07	60003608	6000360E	same

Notes:

1. All addresses shown are expressed as hexadecimal values.

2. Addressing for HI-6131 uses device internal addresses. Bus addressing for HI-6130 is offset by chip select base address 0x60000000 and microprocessor uses byte addressing so all address offsets are doubled. (The LSB becomes upper/lower byte select for each word.)

3. Memory allocations are shared for undefined and reserved mode code commands, and unimplemented subaddress commands, These commands are grouped by like requirements, and share common RAM resources (bit bucket).

4. For messages needing an odd number of words, an extra "pad" word is added so the next buffer begins at an even address.

5. Subaddresses using circular buffer Mode 1 are followed by a 32-word overrun buffer, in case a 32 data word receive command arrives with just one location remaining before "buffer full" attainment.

REMOTE TERMINAL RT2 MEMORY MAP FOR HI-6130 AND HI-6131 APPLICATION DEVELOPMENT BOARD PROGRAM

	dec	hex
Descriptor Table Base Addr	1536	0600
First Buffer Address	16384	4000

Descriptor Table Sector	Device Internal Addr same as HI-6131 Addr		Data Bus Addr Hex HI-6130 Only	
	Start	*End*	*Start*	*End*
Receive Subaddresses	0600	067F	60000C00	60000CFE
Transmit Subaddresses	0680	06FF	60000D00	60000DFE
Receive Mode Codes	0700	077F	60000E00	60000EFE
Transmit Mode Codes	0780	07FF	60000F00	60000FFE

Buffer Assignments for Receive and Transmit Subaddresses

Receive (Rx) Subaddress or Transmit (Tx) Subaddress	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Rx SA1	ping-pong	DPA	34	4000	4021	60008000	60008042	MIW + TT + 32 words
(data pointers A, B and		DPB	34	4022	4043	60008044	60008086	same
broadcast data pointer)		BDP	34	4044	4065	60008088	600080CA	same
Tx SA1	ping-pong	DPA	34	4066	4087	600080CC	6000810E	same
(data pointers A, B and		DPB	34	4088	40A9	60008110	60008152	same
broadcast data pointer)		BDP	4	40AA	40AD	60008154	6000815A	MIW + TT + 2 pad
Rx SA30 and Tx SA30 for data wrap-around	index-0	DPA	34	40AE	40CF	6000815C	6000819E	MIW + TT + 32 words
Rx SA2	index-32	DPA	1088	40D0	450F	600081A0	60008A1E	32 x (MIW + TT + 32 words)
		BDP	34	4510	4531	60008A20	60008A62	
Tx SA2	index-32	DPA	1088	4532	4971	60008A64	600092E2	
		BDP	4	4972	4975	600092E4	600092EA	MIW + TT + 2 pad
Rx SA3	circ1-32	DPA	1088	4976	4DB5	600092EC	60009B6A	32 x (MIW + TT + 32 words)
		pad	32	4DB6	4DD5	60009B6C	60009BAA	pad for overrun
Tx SA3	circ1-32	DPA	1088	4DD6	5215	60009BAC	6000A42A	32 x (MIW + TT + 32 words)
		pad	32	5216	5235	6000A42C	6000A46A	pad for overrun
shared buffer: all unimplemented Rx subaddresses	index-0	DPA	34	5236	5257	6000A46C	6000A4AE	MIW + TT + 32 words
shared buffer: all unimplemented Tx subaddresses	index-0	DPA	34	5258	5279	6000A4B0	6000A4F2	MIW + TT + 32 words
RAM assigned below (MCs)	---	---	142	527A	5307	6000A4F4	6000A60E	
assigned to BC	---	---	248	5308	53FF	6000A610	6000A7FE	BC Msg Data Buffers (excl mode commands)
SA4 not used by RT2								
assigned to MT	---	---	11264	5400	7FFF	6000A800	6000FFFE	IMT Stack or SMT Stacks

Shared Buffer Assignments for Undefined and Reserved Mode Code Commands

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Undefined & Reserved Receive (Rx) Mode Codes Transmit (Tx) Mode Codes	Buffer Method and Data Pointer(s)	Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
shared buffer: undefined Rx MC0 - MC15	index-0 DPA	4	527A	527D	6000A4F4	6000A4FA	MIW + TT, 0 data, 2 pad
shared buffer: undefined Rx MC16, undefined Rx MC18 - MC19, reserved Rx MC22 - MC31	index-0 DPA	4	527E	5281	6000A4FC	6000A502	MIW + TT, 1 data, 1 pad
shared buffer: undefined Tx MC9 - MC15	index-0 DPA	4	5282	5285	6000A504	6000A50A	MIW + TT, 0 data, 2 pad
shared buffer: undefined Tx MC17, undefined Tx MC20 - MC21, reserved Tx MC22 - MC31	index-0 DPA	4	5286	5289	6000A50C	6000A512	MIW + TT, 1 data, 1 pad

Buffer Assignments for Defined Transmit Mode Code Commands MC0 - MC8 (No Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Transmit (Tx) Mode Code Commands, No Data	Buffer Method and Data Pointer(s)	Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Tx MC0	ping-pong DPA	2	528A	528B	6000A514	6000A516	MIW + TT
	DPB	2	528C	528D	6000A518	6000A51A	same
	BDP	2	528E	528F	6000A51C	6000A51E	same
Tx MC1	ping-pong DPA	2	5290	5291	6000A520	6000A522	MIW + TT
	DPB	2	5292	5293	6000A524	6000A526	same
	BDP	2	5294	5295	6000A528	6000A52A	same
Tx MC2	ping-pong DPA	2	5296	5297	6000A52C	6000A52E	MIW + TT
	DPB	2	5298	5299	6000A530	6000A532	same
	BDP	2	529A	529B	6000A534	6000A536	same
Tx MC3	ping-pong DPA	2	529C	529D	6000A538	6000A53A	MIW + TT
	DPB	2	529E	529F	6000A53C	6000A53E	same
	BDP	2	52A0	52A1	6000A540	6000A542	same
Tx MC4	ping-pong DPA	2	52A2	52A3	6000A544	6000A546	MIW + TT
	DPB	2	52A4	52A5	6000A548	6000A54A	same
	BDP	2	52A6	52A7	6000A54C	6000A54E	same
Tx MC5	ping-pong DPA	2	52A8	52A9	6000A550	6000A552	MIW + TT
	DPB	2	52AA	52AB	6000A554	6000A556	same
	BDP	2	52AC	52AD	6000A558	6000A55A	same
Tx MC6	ping-pong DPA	2	52AE	52AF	6000A55C	6000A55E	MIW + TT
	DPB	2	52B0	52B1	6000A560	6000A562	same
	BDP	2	52B2	52B3	6000A564	6000A566	same
Tx MC7	ping-pong DPA	2	52B4	52B5	6000A568	6000A56A	MIW + TT
	DPB	2	52B6	52B7	6000A56C	6000A56E	same
	BDP	2	52B8	52B9	6000A570	6000A572	same
Tx MC8	ping-pong DPA	2	52BA	52BB	6000A574	6000A576	MIW + TT
	DPB	2	52BC	52BD	6000A578	6000A57A	same
	BDP	2	52BE	52BF	6000A57C	6000A57E	same

Buffer Assignments for Defined Transmit Mode Code Commands MC16, MC18 and MC19 (1 Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Transmit (Tx) Mode Code Commands with Data Word	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Tx MC16	ping-pong	DPA	4	52C0	52C3	6000A580	6000A586	MIW + TT, 1 data, 1 pad
		DPB	4	52C4	52C7	6000A588	6000A58E	same
		BDP	4	52C8	52CB	6000A590	6000A596	same
Tx MC18	ping-pong	DPA	4	52CC	52CF	6000A598	6000A59E	MIW + TT, 1 data, 1 pad
		DPB	4	52D0	52D3	6000A5A0	6000A5A6	same
		BDP	4	52D4	52D7	6000A5A8	6000A5AE	same
Tx MC19	ping-pong	DPA	4	52D8	52DB	6000A5B0	6000A5B6	MIW + TT, 1 data, 1 pad
		DPB	4	52DC	52DF	6000A5B8	6000A5BE	same
		BDP	4	52E0	52E3	6000A5C0	6000A5C6	same

Buffer Assignments for Defined Receive Mode Code Commands MC17, MC20 and MC21 (1 Data Word)

These RAM buffer allocations for mode code commands only apply when the application does not use the SMCP option.

Defined Receive (Rx) Mode Code Commands with Data Word	Buffer Method and Data Pointer(s)		Buffer Size *Words*	Device Internal Addr same as HI-6131 Addr *Start*	*End*	Data Bus Addr Hex HI-6130 Only *Start*	*End*	Structures Reserved *MIW = Msg Info Word* *TT = TimeTag Word*
Rx MC17	ping-pong	DPA	4	52E4	52E7	6000A5C8	6000A5CE	MIW + TT, 1 data, 1 pad
		DPB	4	52E8	52EB	6000A5D0	6000A5D6	same
		BDP	4	52EC	52EF	6000A5D8	6000A5DE	same
Rx MC20	ping-pong	DPA	4	52F0	52F3	6000A5E0	6000A5E6	MIW + TT, 1 data, 1 pad
		DPB	4	52F4	52F7	6000A5E8	6000A5EE	same
		BDP	4	52F8	52FB	6000A5F0	6000A5F6	same
Rx MC21	ping-pong	DPA	4	52FC	52FF	6000A5F8	6000A5FE	MIW + TT, 1 data, 1 pad
		DPB	4	5300	5303	6000A600	6000A606	same
		BDP	4	5304	5307	6000A608	6000A60E	same

Notes:

1. All addresses shown are expressed as hexadecimal values.

2. Addressing for HI-6131 uses device internal addresses. Bus addressing for HI-6130 is offset by chip select base address 0x60000000 and microprocessor uses byte addressing so all address offsets are doubled. (The LSB becomes upper/lower byte select for each word.)

3. Memory allocations are shared for undefined and reserved mode code commands, and unimplemented subaddress commands, These commands are grouped by like requirements, and share common RAM resources (bit bucket).

4. For messages needing an odd number of words, an extra "pad" word is added so the next buffer begins at an even address.

5. Subaddresses using circular buffer Mode 1 are followed by a 32-word overrun buffer, in case a 32 data word receive command arrives with just one location remaining before "buffer full" attainment.

BUS CONTROLLER MEMORY MAP FOR HI-6130 AND HI-6131 APPLICATION DEVELOPMENT BOARD PROGRAM

BC Message Blocks
used in application development kit program

Block Number	Command Type	# Block Words	Block Start Addr	Block End Addr	HI-6130 Bus Addr
1	Tx SA *	8	3E00	3E07	60007C00
2	Tx SA *	8	3E08	3E0F	60007C10
3	Rx SA	8	3E10	3E17	60007C20
4	B Rx SA	8	3E18	3E1F	60007C30
5	B Rx SA	8	3E20	3E27	60007C40
6	Tx MC2 ND	8	3E28	3E2F	60007C50
7	Tx MC18 D	8	3E30	3E37	60007C60
8	Rx MC21 D	8	3E38	3E3F	60007C70
RTRT1	RTRT	16	3E40	3E4F	60007C80
RTRT2	B RTRT	16	3E50	3E5F	60007CA0
	available for expansion through end addr...				
		160		3EFF	60007DFE

Corresponding BC Message Data Buffers
used in application development kit program

Number of Words	Buffer Start Addr	Buffer End Addr	HI-6130 Bus Addr
32	5308	5327	6000A610
32	5308	5327	6000A610
32	5328	5347	6000A650
32	5348	5367	6000A690
32	5368	5387	6000A6D0
0	no data	no data	no data
1	1B62	----	600036C6
1	1B55	----	600036AC
32	5388	53A7	6000A710
32	53A8	53C7	6000A750
available for expansion through end addr...			
56		53FF	6000A7FE

* These 2 message blocks are Transmit Subaddress commands to the same subaddress, so use same Tx buffer.

BC Fixed Mode Command Data Word Storage
used in application development kit program

	Mode Code Cmd	# Data Words	Mode Cmd Data Addr		HI-6130 Bus Addr
Receive Mode Code Commands with Data	RxMC 16	1	1B50		600036A0
	RxMC 17	1	1B51		600036A2
	RxMC 18	1	1B52		600036A4
	RxMC 19	1	1B53		600036A6
	RxMC 20	1	1B54		600036A8
	RxMC 21	1	1B55		600036AA
	RxMC 22	1	1B56		600036AC
	RxMC 23	1	1B57		600036AE
	RxMC 24	1	1B58		600036B0
	RxMC 25	1	1B59		600036B2
	RxMC 26	1	1B5A		600036B4
	RxMC 27	1	1B5B		600036B6
	RxMC 28	1	1B5C		600036B8
	RxMC 29	1	1B5D		600036BA
	RxMC 30	1	1B5E		600036BC
	RxMC 31	1	1B5F		600036BE
Transmit Mode Code Commands with Data	TxMC 16	1	1B60		600036C0
	TxMC 17	1	1B61		600036C2
	TxMC 18	1	1B62		600036C4
	TxMC 19	1	1B63		600036C6
	TxMC 20	1	1B64		600036C8
	TxMC 21	1	1B65		600036CA
	TxMC 22	1	1B66		600036CC
	TxMC 23	1	1B67		600036CE
	TxMC 24	1	1B68		600036D0
	TxMC 25	1	1B69		600036D2
	TxMC 26	1	1B6A		600036D4
	TxMC 27	1	1B6B		600036D6
	TxMC 28	1	1B6C		600036D8
	TxMC 29	1	1B6D		600036DA
	TxMC 30	1	1B6E		600036DC
	TxMC 31	1	1B6F		600036DE

BC Instruction List Addresses
used in application development kit program

Op Code #	Op Code Addr	Msg Block called	HI-6130 Bus Addr
0	1B70	op WTG	600036E0
2	1B72	1	600036E4
4	1B74	op WTG	600036E8
6	1B76	2	600036EC
8	1B78	op WTG	600036F0
10	1B7A	3	600036F4
12	1B7C	op WTG	600036F8
14	1B7E	4	600036FC
16	1B80	op WTG	60003700
18	1B82	5	60003704
20	1B84	op WTG	60003708
22	1B86	6	6000370C
24	1B88	op WTG	60003710
26	1B8A	7	60003714
28	1B8C	op WTG	60003718
30	1B8E	8	6000371C
32	1B90	op WTG	60003720
34	1B92	RTRT1	60003724
36	1B94	op WTG	60003728
38	1B96	RTRT2	6000372C
40	1B98	op WTG	60003730
42	1B9A	2	60003734
44	1B9C	op JMP	60003738
46	1B9E		6000373C
48	1BA0	*Execute op codes can call Message Blocks in any order!*	60003740
50	1BA2		60003744
52	1BA4		60003748
54	1BA6		6000374C
56	1BA8		60003750
58	1BAA		60003754
60	1BAC		60003758
62	1BAE		6000375C
available for expansion through end addr...			
142	1BFE		600037FC

Notes:
1. Command Types: SA = Subaddress cmd, MC = Mode Code cmd, ND = no data, D = with data, B = broadcast.
2. All 4-digit hexadecimal addresses refer to the internal IC address, equal to the address used by HI-6131 SPI.
3. The HI-6130 Bus Address = ARM MCU chip select base addr 0x60000000 + 2 x (feature's IC address)

MISCELLANEOUS RAM STRUCTURES NOT ALREADY LISTED

RAM Structure	Start Address	End Address	Number of Words
Interrupt Log Buffer	0x0180	0x01BF	64
Bus Controller General Purpose Queue	0x00C0	0x00FF	64
Bus Controller Call Stack	0x0054	0X005B	8
RT1 Temporary Receive Buffer	0x01C0	0x01DF	32
RT2 Temporary Receive Buffer	0x01E0	0x01FF	32
RT1 Command Ilegalization Table	0x0200	0x02FF	256
RT2 Command Ilegalization Table	0x0300	0x03FF	256
SMT or IMT Message Filter Table	0x0100	0x017F	128
SMT or IMT Address List	0x00B0	0x00B7	8
SMT Command Stack	0x5400	0x5FFF	3072
SMT Data Stack	0x6000	0x7FFF	24577
IMT Combined Stack	0x5400	0x6400	6400