IOB6120

SBC6120 Expander

Updated 5/25/2003 1:11 PM

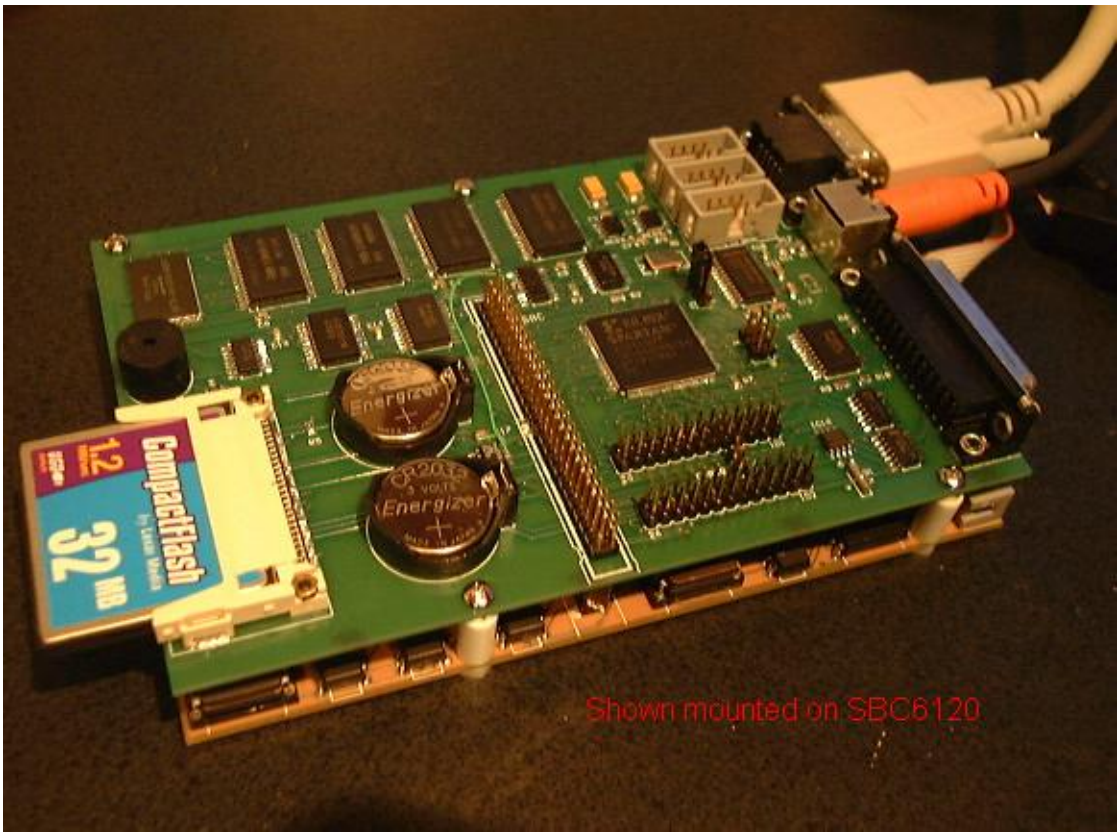
By Jim Kearney

Thanks to Bob Armstrong for many consultations and, of course, the SBC6120 project.

Table Of Contents

Overview

The IOB6120 is a board that expands the Spare Time Gizmos SBC6120. When the two boards are combined, they form a unit only 6.2” deep, 4.2” wide and 1.3” high, yet one that implements a complete PDP-8 system including a VT52 terminal, multiple serial ports, a printer port and a number of other peripherals. On top of all that, the hardware is programmable and can emulate or implement a wide variety of other hardware.



Features

- **CompactFlash™** interface
- **2MB nonvolatile ramdisk**
- **VT52 terminal** with output to a standard VGA monitor and input from a standard PS/2 PC keyboard. The VT52 can be set as the console device by a jumper, and the SBC6120 serial port becomes another SLU.
- **3 serial ports** (SLUs) compatible with DEC KL8JA (RS232C)
- **Printer Port** compatible with DEC LC8E with buffered Centronics parallel output
- **Crystal Clock** DEC compatible timer device
- **36 digital I/O pins** that can be individually configured as inputs or outputs
- **Complete Flexibility** due to its FPGA implementation; IOB functions may be enhanced, removed, replaced or added. For example additional serial ports or a front panel interface could replace some of the digital I/O pins. Using free Xilinx VHDL synthesis software, the 100K-gate FPGA's programming can be altered and tested. Currently, only 68% of the device is used for all the above functionality.
- **Flash ROM** used to store OS extensions and FPGA programming; 128-384KB free for other uses.
- **Clock/calendar** (optional) battery-backed year/month/day hour:minute:second clock
- **Speaker** (audible bell)
- **Enhancements to BTS6120** (see below)

BTS6120 Enhancements

These are implemented in the revised SBC6120 EPROM set, and are useful even without an IOB6120. Note that due to the increase in size of the code, 27C256's or 27C512's must be used rather than 27C64's.

- **Persistent Partition map** - if ramdisk memory is installed via an IOB6120 or other hardware (e.g. the Spare Time Gizmo's ramdisk board), the partition map is saved and restored from it.

- **Partition Compare (PE)** command - checks two partitions for equality, for example after a PC (partition copy) command.

- **IDE boot unit** - in previous versions, the Boot command would always boot from partition 0. This has been changed to boot from the partition mapped as unit 0 using the PM command.

- **Extension ROM support** - a system for extending BTS6120 has been added. When the system starts, a check is made for an extension ROM, and if found, it is installed. The extension ROM can add commands to BTS6120, add function calls for application program use, or modify (hook) existing functions in the EPROMs.

- **Flash Download (FL) command** - if extension ROM as described above exists, it can be initialized by downloading a file over the serial port. Currently only 28Fx00 ROMs are supported, but 29Fx00 types could be used with minor changes.

- **Disassembler** - adds a new X command to disassemble from memory, and enhances the TRace command to display mnemonics as you step through a program.

- **SCope command –** adds a new SC command that changes the function of the DEL key to a backspace, much like OS/8's SET TTY SCOPE.

Installation

Set Up and Use

The following steps should bring up your IOB6120.

Install the new EPROMs and GAL. The GAL to be replaced is U12, which is the one away from the board edge. Power on the SBC6120 and make sure the monitor still comes up on a serial terminal.

With the power off, install the IOB6120 board. Place a jumper block on J3 pins 1-2 and power up. The monitor should come up as before, possibly with a 'ROM: Error xx' message if the Flash ROM has been programmed already.

If the monitor does not start properly, try pressing ESC on the serial terminal during the memory test – just after reset while only the two left POST LEDs are lit. This will prevent BTS6120 from linking in the extension ROM at all. If it still does not work, there is a fault on the IOB6120 that interferes with the SBC6120's bus – check for solder bridges and bad connections.

Using the same terminal download settings as are normally used to download to the SBC6120 (see the manual for details), enter the FL command and answer Y to the prompt, then start downloading the FLASH.DL file from the software distribution directory. If the message '?No Flash' appears, or there is an error 1xxx-3xxx that aborts the download, there is a problem in the IOB6120 in the decoding, buffering or memory section. The most likely cause is that the flash ROM is not soldered down properly – it might help to apply some liquid flux and reflow all the pins.

Remove the jumper from J3 and reset the SBC. New devices should appear in the startup banner. If nothing changes, or 'ROM: Error 2' appears, there is a problem in the Flash ROM's connections.

J3 Jumper Summary	
All open	Use SBC serial port as console, FPGA enabled
2-3 shorted	Use VT52 as console, FPGA enabled
1-2 shorted	Disable FPGA (error 4 will be displayed)

Other possible error messages at this point are:

ROM: Error 3	BTS6120 out-of-date
ROM: Error 4	FPGA INIT signal not ready
ROM: Error 5	FPGA did not configure
ROM: Error 6	FPGA SKIP connection bad
ROM: Error 7	FPGA C0 connection bad

ROM: Error 8	FPGA C1 or DX0-3 connection bad
ROM: Error 9	FPGA data bus connection bad

If one of these errors appears, the FPGA's board connections are probably at fault. Again, check for solder bridges, apply liquid flux and reflow its pins.

Otherwise, the IOB6120 should now be operational! Connect a VGA monitor and PS/2 keyboard and install the jumper at J3 2-3 and reset. The monitor banner should now be on the VGA display, and the PS/2 keyboard active.

A standard Centronics printer can be connected to the printer port and printed to via the standard OS/8 LPT: device.

The three serial ports can be used by copying the SERIAL.PA driver, editing it for one of the devices 30/31, 32/33 or 34/35, assembling it and installing it with BUILD. If the console is set to the VT52, the serial port on board the SBC6120 is also usable as device 36/37.

The CF card socket works much like the IDE drive does, except that its partition numbers are 7700, 7701, and so on. To boot from it, use the PM command to set it as unit 0: 'PM 0 7700', then issue the B command. To use it as a secondary drive, just use PM to map one or more other unit numbers to it.

If using OS/8 and the VT52 (or for that matter a terminal emulator), you might wish to use the command 'SET TTY SCOPE' to enable backspacing rather than the default reverse echo deletion.

Error Codes

The FL command may return the following error codes:

1ccc	Error erasing; ccc is the status register, see 28F400 datasheet
2ccc	Error writing; " " " "
3ccc	Error verifying; ccc is the offset in the current block

CompactFlash operations may return the following error codes:

7777	No card inserted
7776	Timeout waiting for card ready
0ccc	CF returned error register, see CF specifications

Connectors

The VGA, PS/2, printer and three serial connectors all follow the standard pinouts as used in most equipment. The serial ports are labeled SER1, SER2 and SER3 starting from the edge of the board; SER1 is devices 34/35, SER2 30/31 and SER3 32/33.

The expansion connectors provide access to power, the 36 digital I/O lines, and the data bus. The exact pinouts can be found on the schematics and in the VHDL module iob.vhd.

There is a two pin connector just behind the PS/2 socket that connects (via the RS232 level shifter) to the RTS and CTS pins on the SER1 connector. If hardware handshaking (or perhaps paper tape control signals) is desired, these could be wired to two of the digital I/O lines and the appropriate FPGA programming changes made. The serial implementation (KL8JA.vhd) already has RTS/CTS support, but it is not currently bound to any pins.

For faster FPGA programming during development, there is a 6 pin JTAG connector near the FPGA. A 3.3V compatible JTAG programmer can be attached there and a new FPGA configuration file downloaded directly at a high bit rate rather than going through the flash procedure. The pinout of this connector is:

1:GND 2:TMS 3:TDO 4:TCK 5:3.3V 6:TDI

J3 has already been discussed, but for the record, pin 2 is GND, pin 1 is the FPGA INIT line (which, if grounded, prevents it from configuring itself), and pin 3 is an input pin to the status register that is used to determine whether to make the VT52 terminal the console device.

Break

There is a jumper on the SBC6120 that connects the serial break detect to the CPREQ line on the CPU, which allows one to break into the monitor by pressing a key. The IOB6120 can also control this line if the appropriate connections are made. The cleanest way to do this is to install a two pin 0.1" socket on the SBC6120 and a long two pin header on the matching pads on the IOB6120, so that a jumper can still be used on the top side of the IOB6120 if a serial break is desired.

The PS/2 keyboard firmware pulses the CPREQ when the Break key is depressed, so the same functionality is available when using the VT52 terminal.

Power Consumption

With the IOB6120 installed and a PS/2 keyboard connected (which draws its power from the host), the system requires about twice the power that the bare SBC6120 does:

Phase	***SBC6120 Alone***	***SBC610+IOB6120***
Startup	370 mA	510 mA
Steady	200 mA	390 mA

Features in Depth

Hardware Description

FPGA-based

Much of the functionality of the IOB6120 is implemented as logic within a largish Field Programmable Gate Array. This device, a Xilinx Spartan II XC2S100, contains a large number of random logic and RAM cells. All of the emulated DEC devices (VT52, serial ports, and so on) are written in a hardware description language (VHDL) and compiled into a 'bitstream'. This bitstream is stored in an on-board flash ROM and downloaded by the CPU during system startup. Since the hardware is 'soft', and the bitstream is stored in rewritable storage, modifications and extensions can be accomplished without physical redesign.

The VHDL is compiled and synthesized into hardware using the free Xilinx WebPack software. The project file is iob1.npl in the FGPA subdirectory.

Addressing

The IOB6120 is both memory and I/O mapped. The SBC6120 decodes a special extended memory space for its ramdisk option. The IOB6120 uses this memory space for its ramdisk hardware, too, but also uses it for accessing the flash ROM, the CompactFlash socket, and programming the FPGA. The FPGA is connected only to the data bus, and so must be accessed using IOTs.

CompactFlash™ interface

The CompactFlash (CF) socket is mapped directly into the extended memory space and uses 8 addresses (within a decoded 1MB area!). The socket is wired in the Common Memory Model configuration; in this mode it is 8-bit compatible and can be hot-swapped, unlike the True IDE configuration that's often used.

The software merges the CF card into the IDE partition space, so that all BTS6120 commands (DF, DD, PC, etc.) operate on both the CF and IDE drives, and only one driver is required for application software (e.g. OS/8).

In the FLX8 subdirectory, there is a Win32 port of Bob Armstrong's FLX8 that can directly access CF cards (and other block devices). This makes it very easy to move files between the PC and OS/8. See the FLX8-W32.TXT file for details.

2MB nonvolatile ramdisk

Because the IOB6120 precludes the use of the Spare Time Gizmos ramdisk (due to its use of the extended memory space), the nonvolatile SRAM circuitry is also implemented on this board.

VT52 terminal

A complete VT52 compatible terminal which displays on a standard VGA display and uses a standard PS/2 PC keyboard is implemented in the FPGA. To the CPU it looks like a standard console serial port. In fact, in order to be compatible with existing software, the input and output emulation slows the data rate down to the equivalent of a 56Kb serial port. The display hardware is actually capable of displaying text at a rate much faster than the CPU can generate characters.

It should also be possible to drive a 'bare' LCD panel directly, but this is a future project. The source code for the VT52 consists of the files VT52.vhd and its subentities in the FPGA subdirectory.

The device number of the keyboard and VT52 can be set via software. By default, they are 36 and 37, but booting with J3 set to 2-3 will set it to 03 and 04 (and put the SBC6120 console port at 36 and 37).

See the appendices for a list of ESC codes and the PC keyboard mapping.

Serial Ports (SLUs)

3 KL8JA serial boards with RS232 output are emulated, including double-buffering and adding software selectable baud rates from 50 to 115K. The format is fixed at 8 data bits and 1 stop bit, but this can be changed in the VHDL. Likewise, no flow control or 'reader run' is implemented, but some of the digital I/O lines could be redirected to this use by writing a bit of VHDL.

Additional serial ports could be instantiated on the digital I/O pins if desired, and only the appropriate RS232 or current loop level shifters would need to be added.

Printer Port

The printer port is very simple. A set of TTL compatible buffers interfaces between a standard DB25 printer connector and the FPGA. The FPGA implements some IOTs that look like a KL8E printer interface and convert the signals to the standard Centronics definition. The printer port is bi-directional, but the current implementation makes no use of that.

Crystal Clock

The crystal clock is also quite simple. It merely raises its flag at 100Hz, and provides the standard clear flag, set interrupt mask, and skip on flag IOTs.

Digital I/O

36 pins (3 words) of digital I/O pins are implemented as the default use of the expansion ports (the two 26 pin connectors). They are controlled through 4 IOTs. The first IOT selects which word one of the other IOTs will operate on. The next two IOTs control the function of each bit in the word: '1', '0', or 'Z' (not driven). The last IOT reads the current state of the pin. Thus, each pin may independently be used as an output ('0' and '1'), an open-drain input/output ('0' and 'Z') or an input ('Z').

Flash ROM

A 256KB or 512KB flash ROM is used to store the initialization code, BTS6120 extension code, and the FPGA configuration data. Currently only the first 128KB is used for this and the rest of the ROM is available for use. The new FL command allows downloading data into the Flash, and a new ROM function call can access it.

Clock / Calendar

Optionally a Dallas/Maxim DS1302 I^2C clock/calendar chip and its associated crystal can be installed. The FPGA implements a set of IOTs that perform register reading and writing through the I^2C interface. The backup power supply for the DS1302 is BATB (the one away from the edge of the board), so if only one battery is installed it should be this one. The battery should be replaced when power is on to avoid losing the current settings in this case.

Speaker (audible bell)

A single bit tone generator implemented in the FPGA drives a small speaker, and is (currently) used for the VT52 bell.

Interrupts

All of the emulated DEC devices generate interrupts just like the original hardware did, so software using them should operate correctly. The documentation available was not always concise, so there may be areas that need work…

Software

CompactFlash™ interface

During system initialization, the extension ROM initialization function is executed (see below). One of the things it does is intercept the IDE disk read and write functions and redirect them to the extension ROM code. This code splits the partition space into 2 parts. The first part, partitions 0 through 7677 (octal) continues to access the IDE interface on the SBC6120. The second part, partitions 7700 through 7777, access the CF socket. To applications such as OS/8, this is completely transparent since they operate via unit numbers which are limited to 8 devices. The BTS6120 PM (Partition Map) command establishes which partition corresponds to which unit number, so an application may access any mix of IDE and CF partitions. This changes lowers the maximum accessible storage on an IDE device to 8064MB (4032 partitions), and the maximum accessible CF card storage is 512MB (256 partitions). For a PDP-8 of course this is close to infinity.

Persistent Partition map

If ramdisk memory is installed via an IOB6120 or other hardware (e.g. the Spare Time Gizmo's ramdisk board), the partition map is saved and restored from it. The ramdisk address to block mapping leaves a small amount of memory unused (16KB per MB), and a very small part of this is used to store the partition map.

Partition Compare (PE) command

This command compares the checksums of each block from two different partitions. Its only use really is to verify that a Partition Copy worked properly, and it serves as hardware check for the CF card socket.

IDE Boot Unit

In previous versions of BTS6120, the secondary bootstrap was always read from partition 0. Since the bootstrap then used the ROM call and thus the partition map for the rest of the boot, all was well. However, now there is a possibility that there is no drive connected to the IDE connector and partition 0, but there is a CF card. Therefore the change was made to use the partition mapped to unit 0 as the secondary bootstrap location.

Extension ROM support

BTS6120 has been extended with the ability for a program (loaded into panel RAM) to add commands and ROM function calls, and to intercept certain internal functions (such as disk I/O). During startup, a specific area of extended memory (04000000-04007777) is

checked for a signature and a valid checksum, and if those succeed, the code found there is copied to field 2 of the panel RAM and executed.

Flash Download (FL) command

In order to easily load and update software for expansion boards, particularly the IOB6120, a command was added to BTS6120 that downloads binary from the console into the Flash ROM. The most common use of this command is to install the FLASH.DL file, but it may also be used to store arbitrary data into the parts of the flash that are not used by the firmware.

The device used is the 28F400, available from many sources, all with the same programming commands, so this command should be able to program any vendor's IC. Note that the 29F400 will also work on the PCB, but the FL command will need to be modified to accommodate its different programming commands. The download uses the same basic format as the DD/DL commands do, except that each item to be programmed is a byte rather than a 12-bit word. For every 256 bytes downloaded a checksum is verified, and after programming each 256 bytes the ROM is verified against the source data.

A Flash ROM is laid out in blocks, and an entire block is erased at a time. When erased, the value FF is read. Blocks vary in size, even in the same device (boot blocks are smaller than the main blocks in some types, for example), but are generally multiples of 4KB. The FL command erases as it goes, and it does this by, every 4KB, checking to see if the next 4KB are erased. If not, an erase command for this block is issued. In the LH28F400VHBE-BL currently used in the IOB6120, the first 8 blocks are 4KB and the last 7 are 32KB. Therefore to be able to update just a section of the flash, the downloads should be arranged to fit this layout. The first two 4KB blocks and the first 4 32KB blocks are used by the standard firmware.

Disassembler (X/XP/TR) command

Two new commands are added – X and XP. They are almost the same; the difference is that X disassembles from main memory and XP from panel memory. They require a parameter that is either a single address or an address range. For memory reference instructions, the word at the referenced address is displayed after the instruction, and for indirect references the word referred to by that one is also displayed. Note that, if the DF register changes during execution, these indirect references may not always be accurate.

The disassembler is also linked to the TR command, so that when single stepping through a program the mnemonics will be shown.

Date (DA) Command

The current date and time may be displayed using the DA command in BTS6120. The same commands sets the date and time, just type the new, complete date and time after DA in the same format that it's displayed in. By default, the display format is U.S. standard MM/DD/YY HH:MM:SS, but this may be changed by editing the ROM source file, re-assembling and downloading it. If the FPGA is not enabled, this command will not be available.

Scope (SC) Command

The SC command is followed by a number. If the number is 0, the DEL key will echo characters typed in reverse order (like the default mode of OS/8, but without the slashes). If a non-zero parameter is given, the DEL key will act the same as Backspace (ctrl-H). This is parallel to the OS/8 SET TTY SCOPE command. This value is persistent if you have NVRAM installed.

Memory Move Extended ROM function

A new version of the memory move function has been added that has the capability of accessing the Flash ROM and NVRAM memory.

```
PR0
 0011              / copy memory subfunction
 pddf              / source DAR, field and memory space
 pddf              / destination DAR, field and memory space
 <address>         / source address
 <address>         / destination address
 <word count>      / number of words to be transferred
```

The source and destination field words (pddf) each contain a flag in bit 0, the ramdisk Disk Address in bits 1..8 (or a flag in bit 1), and the field number in bits 9..11

The flags are interpreted as

bit 0 = 1 when the address is in the ramdisk, otherwise

bit 1 = 1 when the address is in panel memory, otherwise
the address is in main memory

The last word of the argument list is the number of words to be copied - a value of zero copies 4096 words.

The Disk Address value is the upper 8 bits of the address, and the field value is 1 for the Flash ROM, and 0,2,4 and 6 for the SRAMs. Other field numbers are used for internal functions.

User data may be downloaded into the Flash ROM starting at DAR 32 and accessed using this function.

Kermit-12

Included in the OS8 subdirectory is a copy of Kermit-12 that has been modified for the IOB6120. It is configured to work with the serial port at device 30/31, which is the center serial connector on the IOB. The baud rate is set at 19200, but can be changed by editing the source and reassembling (or by patching location 7).

Note that the source files in the OS8 subdirectory are plain ASCII while the binaries are in image format. They can be transferred into a .VM or .IDE file using FLX8 and downloaded using the BTS6120 DL command, or directly to a CompactFlash card using FLX8-W32 under Windows:

FLX8>MOUNT <file>/ID01 (or '<device>/ID01/PHYS')
FLX8>WRITE OS8*.PA/ASCII
FLX8>WRITE OS8*.SV/IMAGE
FLX8>DISMOUNT

On a Windows machine with a CF card reader installed, it's probably easier to copy files using a card, rather than using Kermit-12.

Changing or Extending the IOB6120

The extension ROM firmware is located in the ROM subdirectory, and is assembled using Bob Armstrong's PALX assembler. If the IOB6120 directory tree is installed within the BTS6120 directory, the makefiles should find the assembler and other tools without any changes.

Running the makefile in the IOB6120 directory will rebuild the extension ROM and generate three downloadable .HEX files:

flash.dl	both the extension code and the FPGA code
cflash.dl	only the extension code
fflash.dl	only the FPGA code

Note that the extension ROM startup only downloads the first 2688 words (4KB) into RAM, so if your modified code extends to address 25200, you will need to (a) split your additions into a new field, and (b) download that new field from the first extension ROM field.

If changes to the extension ROM prevent BTS6120 from operating properly, pressing ESC on the serial terminal during the memory test – just after reset while only the two left POST LEDs are lit. This will prevent BTS6120 from linking in the extension ROM at all. Debugged code can than be downloaded using the FL command.

The FPGA code is synthesized using the free Xilinx WebPack (http://www.xilinx.com/ise). After registering and downloading this software (the accompanying ModelSim is also recommended), just open the project in the FPGA directory from the Project Navigator and double-click on the 'Generate Programming File' process. This brings the code file, iob.bit, up to date. Run the makefile in the IOB6120 directory to update the download .HEX file, as with the extension ROM code.

For example, to change the first serial port's address, open (double-click) the iob_cfg.vhd file and find the lines starting 'constant UARTxBASE …'. Edit the addresses, and then right-click on the 'Generate Programming File' process and select 'Re-run All'. Then run the makefile and download the fflash.dl file to the IOB6120.

Reprogramming the FPGA 'on the fly'

During startup the FPGA program is downloaded from the ROM image, but for special purposes it can be reprogrammed dynamically. For example, a program that has a special piece of virtual hardware could store a custom FPGA programming bitstream in a file and download it as part of its startup.

The following steps are necessary:

1. Initate FPGA reset:
   IOT 6423 with AC=101010101011
   IOT 6425 with AC=001010011100

2. Wait for FPGA ready:
   Using the extended memory move ROM function call, read location 7000xxxx and wait until AC10 is ‘1’.

3. Transfer the data using the extended memory move ROM function call from a memory area containing an 8-bit byte in each word to location 500xxxx, with up to a full field (4KW) transferred each call.

4. Verify that the FPGA accepted the bitstream without error by reading location 7000xxxx and checking that AC11 is ‘1’.

Assembly

Tools

You will need at least the following tools:

- Temperature controlled soldering iron (e.g. Hakko 936 ESD)
- Fine tip, 1/64" (e.g. Hakko 900M-T-I)
- Flux No-Clean or RMA
- Fine Solder, 0.010" or 0.015"

  - And/Or -
- Solder paste in fine gauge dispenser (syringe)
- Fine solder wick (e.g. No Clean Soder Wik)
- 7.5x – 10x Loupe and optionally, for the medium-sized parts, a 3X - 5X headband magnifier
- Good light source, e.g. a gooseneck lamp
- Fine probe (Dental probe)
- Printed Circuit Board holder
- Anti-Static mat and/or other method of preventing ESD
- Flux cleaner
- Fine bristle brush

Of course one can get much more sophisticated equipment than this, for example hot air tools, solder paste dispensers, inspection microscopes, etc., but the above suffices with care and a little practice. It's possible to get many of the tools and supplies in a kit, for example the Chipquik SMD 2000.

Board Preparation

Before soldering, buff the board with a sheet of paper or paper towel to remove any oxidation on the pads. You may wish to wipe it with alcohol as well.

Assembly process

For hand assembly, the best way to proceed is to mount the thinner and finer-pitch components first so as to maximize our access to their pins.

See the parts layout diagram in the Appendix. Be ABSOLUTELY CERTAIN that you mount the IC's are the right way 'round. The silkscreen indicates pin 1 or the pin 1 side of every IC. Also note that C1 and C2 are polarized capacitors, and there is a band marked on the capacitor and on the silkscreen.

The recommended order of assembly is IC1 and IC12 first (Flash ROM and FPGA), followed by the chip resistors and capacitors and oscillator, the CompactFlash socket, and then the half-pitch components (the rest of the IC's), and finally the through-hole components.

The actual process of soldering SMT parts is well covered by the links below.

Make sure to get the IC's placement as exact as possible before soldering them down by tacking two corner pins and adjusting the position until ***all*** the other pins line up (the loupe is invaluable for this). Then touch the soldering iron's tip to each pin either until the solder paste melts (if using that), or the fine solder that you're applying does. Take care not to use too much solder, or let the tip get too 'wet' or solder bridges will result. If they do appear, use the fine solder wick to clean them up. Run the tip of the soldering iron back and forth a couple of times on each pin between the leg and end of the pad. After soldering each part, inspect it with a loupe under good lighting. Each pin should have a good solder fillet. Run the dental pick with a little force across the pins and (a) look for pins that move, and (b) listen for a different 'twang' that indicates a cold solder joint.

If installing the clock/calendar option, note that there are pads to solder a hold-down wire over the crystal. Glue could also be used.

For initial check out, it's not necessary to mount the connectors, except for the connector to the SBC.

Once the parts are mounted, wash the board by immersing it in flux cleaner and brushing along the leads of the IC's, taking special care with the fine pitch parts to make sure no globs of solder and/or paste remain. Inspect the board again carefully under light and magnification for shorts and unsoldered pins.

Follow the set up and testing procedure in the *Installation and Use* section.

Links

Parts

http://www.digi-key.com/
http://www.mouser.com/

Tools

http://www.engineeringlab.com/
http://www.contacteast.com/
http://www.chipquik.com/
http://www.micro-tools.com/

SMT Soldering Tutorials

http://www.circuittechctr.com/guides/7-5-1.htm
http://www.circuittechctr.com/guides/7-3-1.htm
http://www.bdmicro.com/smt/
http://www.cygnal.com/appnotes/an014.pdf

Appendix A - Parts List

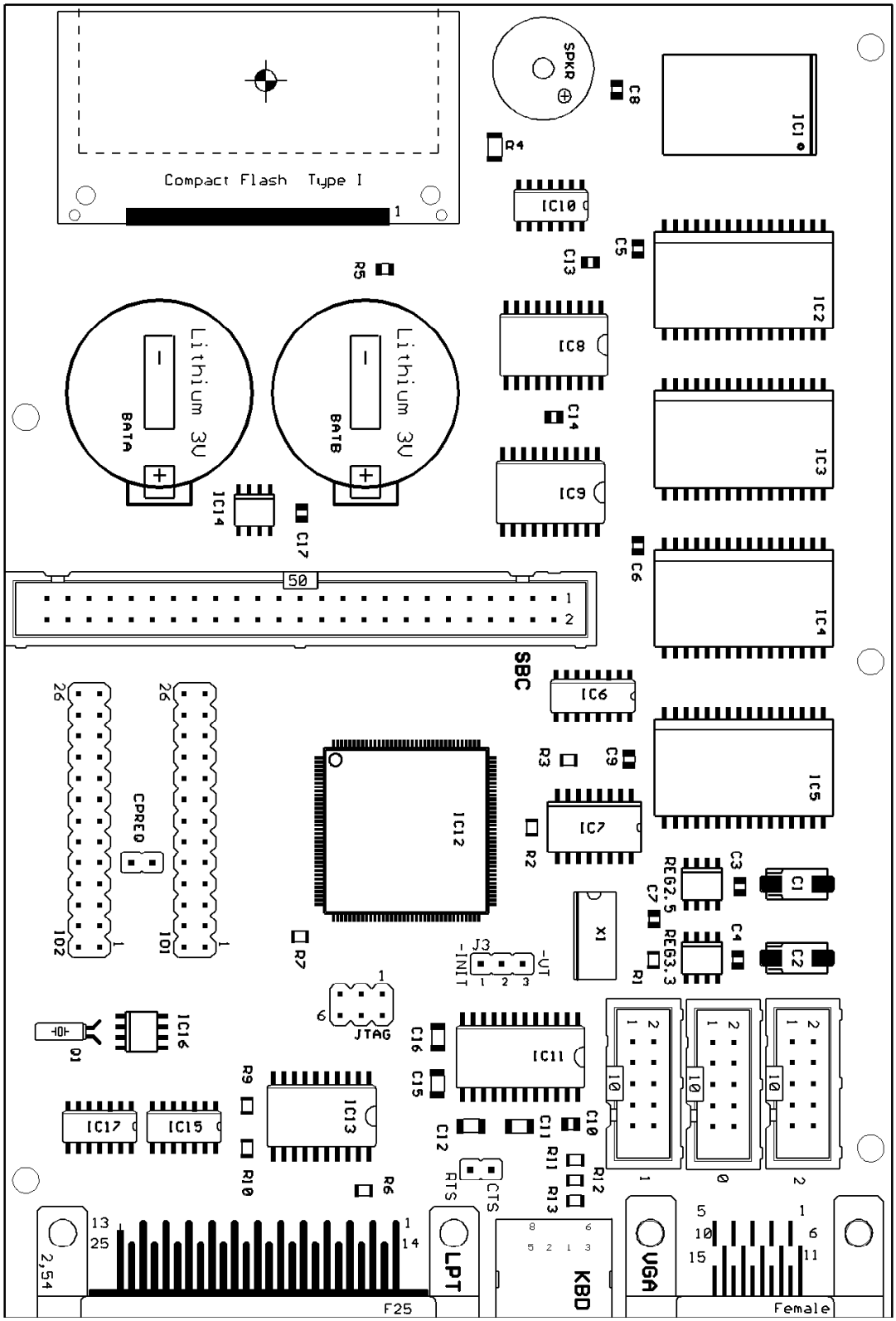
Qty	Group	ID	Part Number	Description	Digi-Key #
1	C	IC1	LH28F400BVHE-BL	256/512KB flash ROM	
4	R	IC2, IC3, IC4, IC5	TC554001AF	low power 512KB SRAM	TC554001AF-70L-ND
1	C	IC6	74ACT138M	3-to-8 decoder	296-4255-5-ND
1	C	IC7	74AHC367NSR	Hex buffer	296-12898-1-ND
1	C	IC8	74ACT373DW	Octal latch	296-4421-5-ND
2	C, P	IC9, IC13	74ACT245SC	Octal bi-directional buffer	74ACT245SC-ND
1	C	IC10	74ACT32D	Quad OR	74ACT32SC-ND
1	S	IC11	MAX208CWG	RS232 Transciever 4/4	MAX208CWG-ND
1	C	IC12	XC2S100-5TQ144	Spartan II FPGA	122-1229-ND
1	R	IC14	BQ2201SN / MXD1210CSA	Nonvolatile RAM controller	296-9370-5-ND
1	P	IC15	74ACT05M	Hex OC inverter	296-4252-5-ND
1	P	IC17	74ACT04D	Hex inverter	296-1065-5-ND
1	C	REG2.5	TPS77725D	2.5V regulator	296-2786-5-ND
1	C	REG3.3	TPS77733D	3.3V regulator	296-2787-5-ND
5	C	R1, R2, R3, R5, R7	R-US_M0805 10K	Resistor 0805 package	
3	V	R11, R12, R13	R-US_M0805 301	Resistor 0805 package	
1	V	R4	R-US_M1206 100	Resistor 0805 package	
3	C	R6, R9, R10	R-US_M0805 4K7	Resistor 0805 package	
2	R	BATA, BATB	CR2032HOLDER	Coin cell holder	BS-7-ND
2	C	C1, C2	CPOL-US085CS 10u	Capacitor 085C package	
4	S	C11, C12, C15, C16	C-USC1206 0u1	Capacitor 1206 package	
19	C	C3, C4, C5, C6, C7, C8, C9, C10, C13, C14, C17, C100, C101, C102, C103, C104, C105, C106, C107	C-USC0805 0u1	Capacitor 0805 package	
1	F	CF1	CompactFlash Socket	Molex 538-56-5010	WM18505-ND
2	C	IO1, IO2	2x13 0.1" header	Expansion connectors	
1	C	J3	1x3 0.1" header		

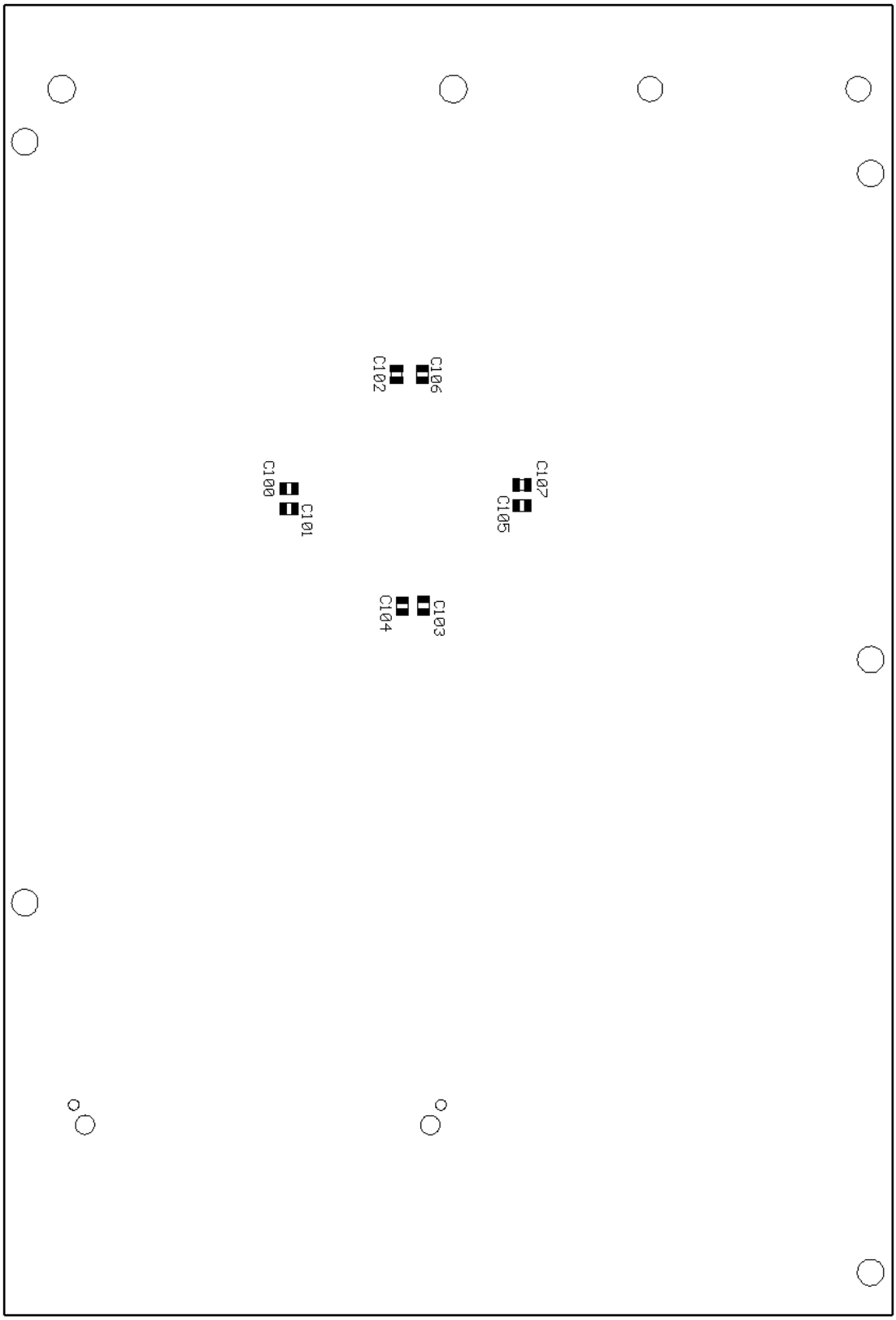
1	C	JTAG	2x3 0.1" header	not mounted	
2	C	CPREQ, HS2	1x2 0.1" long header	not mounted	
1	V	KBD	Mini-DIN 6 female PCB mount	PS/2 Keyboard port	CP-2460-ND
1	P	LPT	DB25 male PCB mount	Printer port	182-25M-ND
1	C	PCB	Printed Circuit Board		
1	C	SBC	2x25 0.1" header	not mounted	
3	S	SER1, SER2, SER3	2x5 0.1" box header	Serial ports	
1	V	VGA	DE15 female PCB mount	VGA port	815RF-ND
1	V	SPKR	KSS1201	Beeper	102-1154-ND
1	C	X1	CSX750FBC29.491200MTR	29.4912MHz 5V SMD Osc.	300-7248-1-ND
Clock / Calendar option					
1	CC	X1	DS1302Z	Dallas/Maxim I2C Calendar	DS1302Z-ND
1	CC	X2	32.768KHz	Watch crystal	X801-ND

Key to Group: C = Core, R = ramdisk, F = CompactFlash, S = Serial, P = Parallel, V = VT52, CC = Clock/Calendar.

Core must be installed; the other groups are optional. If the NV controller (BQ2201) is not installed, some bypass jumpers need to be added.

Appendix B – Parts Layout




C100
C101
C102
C103
C104
C105
C106
C107

Appendix C – VT52 Control and Escape Codes

Code	Action
007 (BEL)	Ring terminal bell.
010 BS	Backspace. Move cursor to the left one character, unless it's already on the left margin.
011 TAB	Move cursor right to next tabstop. Tabstops are every 8 columns.
012 LF	Move the cursor down to next line. If already on bottom line, scroll the screen up one line.
015 CR	Move the cursor to the beginning of the current line.
016 SO	Switch to black-on-white (inverse) mode. Nonstandard.
017 SI	Switch to white-on-black (normal) mode. Nonstandard.
033 101 ESC A	Move the cursor up, unless it's already on the top line.
033 102 ESC B	Move the cursor down, unless it's already on the bottom line.
033 103 ESC C	Move the cursor right, unless it's already at the right margin.
033 104 ESC D	Move the cursor left, unless it's already at the left margin.
033 110 ESC H	Move the cursor Home (top left).
033 111 ESC I	Reverse Linefeed. Move the cursor up. If it's already on the top line, scroll the screen downwards.
033 112 ESC J	Erase from cursor position to the end of the screen.
033 113 ESC K	Erase from the cursor position to the end of the line.
033 131 040+l 040+c ESC Y SP+l SP+c	Move the cursor to an absolute location line l column c, l and c are zero-based.
033 132 ESC Z	Send Identify sequence 033 057 113 (ESC / K)
033 035 ESC =	Switch to alternate keypad mode.
033 036 ESC >	Switch to normal keypad mode.

Appendix D – VT52 PC Keyboard Mapping

The keyboard implementation talks to a standard PC PS/2-style keyboard and generates all the characters and escape sequences that a VT52 keyboard does.

The PC keyboard has a lot more keys than the VT52's, so a lot of the keys are not mapped to any ASCII sequence. In particular, Alt, many cursor movement keys and most of the function keys have no VT52 equivalent.

The keypad is handled a bit differently than the PC's normal usage, too. The unshifted function is as a numeric keypad, and the shift key activates the arrows and DEL key. Num Lock has no function, but the Num Lock LED will light when the keypad is set into 'Alternate Keypad Mode' by the host. The three PF (colored) keys on the VT52 are mapped to F1-F3 on the keyboard.

When the VT-52 Hold Screen Mode is activated from the host, the Scroll Lock LED will light. The Scroll Lock key releases the screen for one line, and Shift- Scroll Lock for one screen.

(Hold Screen mode is currently unimplemented.)

Appendix E – IOT reference

VT52 Terminal Device XX/YY – settable to any device code; normally 03/04 or 36/37		
IOT	AC	Description
6XX0	----------	KCF Clear Keyboard Flag
6XX1	----------	KSF Skip If Keyboard Flag
6XX2	----------	KCC Clear Keyboard Flag, Clear AC
6XX4	Ccccccccccc	KRS OR Keyboard Character into AC
6XX5	----------a	KIE Set Interrupt Enable: a=1 => Interrupts Enabled
6XX6	----------	KRB Get Keyboard Character into AC, Clear Keyboard Flag
6YY0	----------	SPF Set Display Flag
6YY1	----------	TSF Skip If Display Flag
6YY2	----------	TSF Clear Display Flag
6YY4	0000ccccccccc	TPC Display Character, Don’t Clear Display Flag
6YY5	----------	TSK Skip If Keyboard Or Display Flag
6YY6	0000ccccccccc	TLS Display Character, Clear Display Flag

Serial Port (KL8JA SLU) Device XX/YY: 30/31, 32/33, and 34/35		
IOT	AC	Description
6XX0	----------	Clear Input Flag
6XX1	----------	Skip If Input Flag
6XX2	----------	Clear Input Flag, Clear AC
6XX4	Ccccccccccc	OR Input Character into AC
6XX5	----------a	Set Interrupt Enable: a=1 => Interrupts Enabled
6XX6	----------	Get Input Character into AC, Clear Keyboard Flag
6YY0	----------	Set Output Flag
6YY1	----------	Skip If Output Flag
6YY2	----------	Clear Output Flag
6YY3	-------rrrrr	Set Baud Rate (See table below for values)
6YY4	0000ccccccccc	Output Character, Don’t Clear Output Flag
6YY5	----------	Skip If Input Or Output Flag
6YY6	0000ccccccccc	Output Character, Clear Output Flag

Code	Baud (bps)
00000	50 (56.25)
00001	75
00010	110 (112.5)
00100	150
00101	300
00110	600
00111	1200

01000	1800
01010	2400
01011	3600
01100	4800
01101	7200
01110	9600
01111	19200
10000	38400

10001	56400

10010	115200

Printer (LC8E) Device 66		
IOT	AC	Description
6660	----------	PSSF Set Printer Flag
6661	----------	PSKF Skip If Printer Flag
6662	----------	PCLF Clear Printer Flag
6664	0000cccccccc	PSTB Print Character without clearing flag
6665	----------a	PCIE Set Printer Interrupt Enable a=1 => Interrupts Enabled
6666	0000cccccccc	PCLB Print Character and Clear Flag

Crystal Clock Device 13		
IOT	AC	Description
6135	----------a	CLLE Set clock interrupt mask. a=1 => interrupts enabled.
6136	----------	CLCL Clear clock flag
6137	----------	CLSK Skip on clock flag

Digital I/O pins Device 41		
IOT	AC	Description
6414	0000000000bb	Set bank to bb, bb=00, 01, or 10. Bank 1 is IO1-12, bank2 IO13-24, and bank 3 IO25-36.
6415	dddddddddddd	Configure the pins in the current bank: d=0 => push/pull output, 1 => Open-Drain or Input. Default = 1.
6416	vvvvvvvvvvvv	Set the output pins in the current bank: v=0 => grounded, 1 => 3.3V or high-Z (depending on the setting from the previous IOT. Default = 1.
6417	----------	Read the current pin value (LVTTL thresholds)

FPGA control and configuration Device 42		
IOT	AC	Description
6412	----------	Set the SBC6120 serial port to the console address, 03/04. AC is cleared.
6413	----------	Set the SBC6120 serial port to the secondary address, 36/37.
6423	101010101011	Safety Unlock, must immediately precede either of the next two IOTs.
6424	kkkkkkvvvvvv	Set the VT52 device codes: kkkkkk for the keyboard and vvvvvv for the video.
6425	001010011100	Initiate FPGA reprogramming.
6426	---------abc	Self-Test Signals: set C0 if a=1, C1 if b=1,

FPGA control and configuration Device 42		

		and SKIP if c=1. AC will be set as follows:
		if ab=00, AC will be unchanged;
		if ab=01, AC will be Ored with 101010101010;
		if ab=10, AC will be 000000000000;
		if ab=11, AC will be 101010101010.

6427	aaaaaaaaaaaa	Self-Test Bus: AC should be the ones complement of the input value aaaaaaaaaaaa.

Clock/Calendar Device 14		
IOT	AC	Description
6140	00001raaaaaw	Issue command r=1 => RAM, r=0 => register. aaaaa => address (see below), w=1 => read, else write. If RAM is addressed, locations 00000-11101 are available for use.
6141	----------	Skip if command complete
6142	----dddddddd	Set data register
6143	0000dddddddd	Get data register

00000	Seconds, in BCD. b7 is clock start/stop, stop=1
00001	Minutes, in BCD
00010	Hours, in BCD. b7 is the 24hr flag. If 0 (12Hr), b6 is am/pm, pm=1.
00011	Day, in BCD
00100	Month, in BCD
00101	Day of Week, in BCD
00101	Year, in BCD
00111	msb=1 write protect