Application Note: Autostarting an IsoMax Application

The Autostart Search

When the IsoPod is reset, it searches the Program Flash ROM for an **autostart pattern**. This is a special pattern in memory which identifies an autostart routine. It consists of the value $A55A, followed by the address of the routine to be executed.

xx00: $A55A
xx01: address of routine

It must reside on an address within Program ROM which is a multiple of $400, i.e., $0400, $0800, $0C00, ... $7400, $7800, $7C00.

The search proceeds from $0400 to $7C00, and terminates when the *first* autostart pattern is found. This routine is then executed. If the routine exits, the IsoMax interpreter will then be started.

Writing an Application to be Autostarted

Any defined word can be installed as an autostart routine. For embedded applications, this routine will probably be an endless loop that never returns.

Here's a simple routine that reads characters from terminal input, and outputs their hex equivalent:

```
: MAIN   HEX BEGIN KEY . AGAIN ;   EEWORD
```

Note the use of `EEWORD` to put this routine into Flash ROM. An autostart routine must reside in Flash ROM, because when the IsoPod is powered off, the contents of RAM will be lost. If you install a routine in Program RAM as the autostart routine, the IsoPod will crash when you power it on. (To recover from such a crash, see "Bypassing the Autostart" below.)

Because this definition of `MAIN` uses a `BEGIN...AGAIN` loop, it will run forever. You can define this word from the keyboard and then type `MAIN` to try it out (but you'll have to reset the IsoPod to get back to the command interpreter). This is how you would write an application that is to run forever when the IsoPod is reset.

You can also write an autostart routine that exits after performing some action. One common example is a routine that starts some IsoMax state machines. For this discussion, we'll use a version of `MAIN` that returns when an escape character is input:

```
HEX
: MAIN2   HEX BEGIN KEY DUP .  1B = UNTIL ;   EEWORD
```

In this example the loop will run continuously until the ESC character is received, then it exits normally. If this is installed as the autostart routine, when it exits, the IsoPod will proceed to start the IsoMax command interpreter.

Installing an Autostart Application

One the autostart routine is written, it can be installed into Flash ROM with the command

address `AUTOSTART` routine-name

This will build the autostart pattern in ROM. The *address* is the location in Flash ROM to use for the pattern, and must be a multiple of $400. Often the address $7C00 is used. This leaves the largest amount of Flash ROM for the application program, and leaves the option of later programming a new autostart pattern at a lower address. (Remember, the autostart search starts low and works up until the *first* pattern found, so an autostart at $7800 will override an autostart at $7C00.) So, for example, you could use

```
HEX 7C00 AUTOSTART MAIN2
```

to cause the word MAIN2 to be autostarted. (Note the use of the word HEX to input a hex number.)

Try this now, and then reset the IsoPod. You'll see that no "IsoMax" prompt is displayed. If you start typing characters at the terminal, you'll see the hex equivalents displayed. This will continue forever until you hit the ESC key, at which point the "IsoMax" prompt is displayed and the IsoPod will accept commands.

Saving the RAM data for Autostart

Power the IsoPod off, and back on, and observe that the autostart routine still works. Then press the ESC key to exit to the IsoMax command interpreter. Now try typing MAIN2. IsoMax doesn't recognize the word, even though you programmed it into Flash ROM! If you type WORDS you won't see MAIN2 in the listing. Why?

The reason is that some information about the words you have defined is kept in RAM[1]. If you just reset the board from MaxTerm, the RAM contents will be preserved. But if you power the board off and back on, the RAM contents will be lost, and IsoMax will reset RAM to known defaults. If you type WORDS after a power cycle, all you will see are the standard IsoMax words: all of your user-defined words are lost.

To prevent this from happening, you must save the RAM data to be restored on reset. This is done with the word SAVE-RAM:

```
SAVE-RAM
```

This can be done either just before, or just after, you use AUTOSTART. SAVE-RAM takes a "snapshot" of the RAM contents, and stores it in Data Flash ROM. Then, the next time you power-cycle the board, those preserved contents will be reloaded into RAM. This includes *both* the IsoMax system variables, and any variables or data structures you have defined.

Note: a simple reset will not reload the RAM. When the IsoPod is reset, it first checks to see if it has lost its RAM data. Only if the RAM has been corrupted -- as it is by a power loss -- will the IsoPod attempt to load the SAVE-RAM snapshot. (And only if there is no SAVE-RAM snapshot will it restore the factory defaults.) If you use MaxTerm to reset the IsoPod, the RAM contents will be preserved.

Removing an Autostart Application

Don't try to reprogram MAIN2 just yet. Even though the RAM has been reset to factory defaults, MAIN2 is still programmed into Flash ROM, and IsoMax doesn't know about it. In fact, if you try to redefine MAIN2 at this point, you might crash the IsoPod, as it attempts to re-use Flash ROM which hasn't been erased. (To recover from this, see "Bypassing the Autostart," below.)

To completely remove all traces of your previous work, use the word SCRUB:

```
SCRUB
```

This will erase all of your definitions from Program Flash ROM -- including any AUTOSTART patterns which have been stored -- and will also erase any SAVE-RAM snapshot from Data Flash ROM. Basically, the word SCRUB restores the IsoPod to its factory-fresh state.

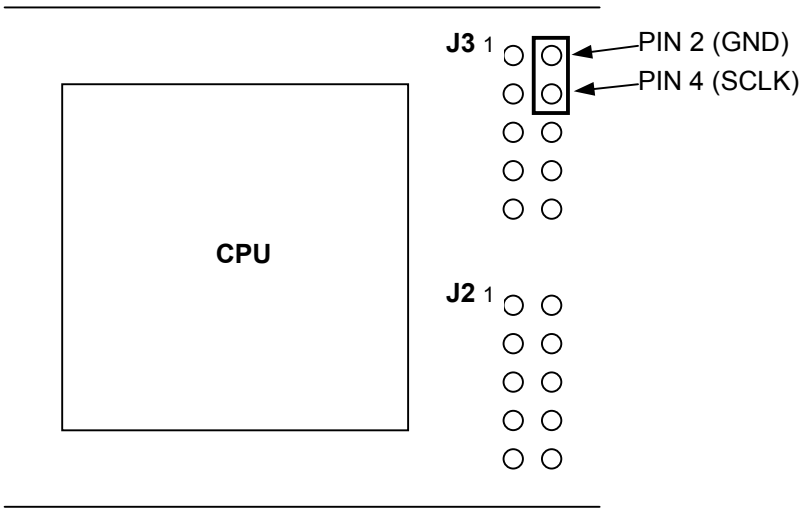
Bypassing the Autostart (Note: this feature will be added on V0.4 or higher)

What if your autostart routine locks up? If you can't get access to the IsoMax command interpreter, how do you SCRUB the application and restore the IsoPod to usability?

You can bypass the autostart search, and go directly to the IsoMax interpreter, by jumpering together pins 2 and 4 on connector J3, and then resetting the IsoPod. You can do this with a common jumper block:

---

[1] To be specific, what is lost is the LATEST pointer, which always points to the last-defined word in the dictionary linked list. The power-up default for this is the last-defined word in the IsoMax kernel.



This connects the SCLK/PE4 pin to ground. When the IsoPod detects this condition on reset, it does not perform the autostart search.

Note that this does *not* erase your autostart application or your `SAVE-RAM` snapshot from Flash ROM. These are still available for your inspection[2]. If you remove the jumper block and reset the IsoPod, it will again try to run your autostart application. (This can be a useful field diagnostic tool.)

To remove your application and start over, you'll need to use the `SCRUB` command. The steps are as follows:

1. Connect a terminal (or MaxTerm) to the RS-232 port.

2. Jumper pins 2 and 4 on J3.

3. Reset the IsoPod. You will see the "IsoMax" prompt.

4. Type the command `SCRUB` .

5. You can now remove the jumper from J3.

Summary

Use `EEWORD` to ensure that all of your application routines are in Flash ROM.

When your application is completely loaded, use `SAVE-RAM` to preserve your RAM data in Flash ROM.

Use `address AUTOSTART routine-name` to install your routine for autostarting. "address" must be a multiple of $0400 in empty Flash ROM; `HEX 7C00` is commonly used.

To clear your application and remove the autostart, use `SCRUB`. This restores the IsoPod to its factory-new state.

If the autostart application locks up, jumper together pins 2 and 4 of J3, and reset the IsoPod. This will give you access to the IsoMax command interpreter.

---

[2] The IsoPod RAM will be reset to factory defaults instead of to the saved values, but you can still examine the `SAVE-RAM` snapshot in Flash ROM.