MC9S08 & backdoor security key

03-16-2007 08:49 PM • 8,005 Views

irob
Contributor V

Hello, all.

Rather than ask rambling vague questions, let me tell you what I want to do. I have EEPROM emulation in Flash working now in my MC9S08QGx. I also have this working on some older MC908QTx (non-S family) boards too.

The great thing about the QT family security mechanism was that the boot vectors doubled as a security code. With my P&E Micro programmer, I could very easily read the contents of the user flash, e.g., all that EEPROM-like user data. The "history" of the board could be read. This has proved to be a great troubleshooting tool.

On the newer 9S08 boards, I don't believe I have this option, at last with P&E tools. I've asked the question over at the P&E forums, and one of their tech support people verified that suspicion.

But I'm very curious about the backdoor security key feature in the S family. I've read this post and this post, but I'm still foggy.

From the datasheet (page 50), it sounds to me like this should be possible:

> ...a secure user program can temporarily disengage security by: 2. Writing the user-entered key values to the NVBACKKEY through NVBACKKEY+7 locations... User software normally would get the key codes from outside the MCU system through a communication interface such as a serial I/O.

So can I do what I want in the 9S08 -- secure the flash, yet read back the stored user data later with the backdoor key?

Labels: General

7 Replies

03-16-2007 11:50 PM • 633 Views

peg
Senior Contributor IV

Hi Rob,

The old HC08 security method was handy in that it had a fixed, externally accessable entry method. You just needed to know the code do the procedure and your in.
This allowed the tool makers (e.g. P&E) to implement security bypass into their tools. Even so far as to extract the code from an S19 for you. All you needed was the source code or FLASH image.

The S08 while the entry mode is fixed it CANNOT be executed from outside. The procedure must be carried out from inside by code YOU write. This allows the level of security to be enhanced greatly, by you. This however means it is now impossible for the tool makers to provide any backdoor support as they don't know what you have done.

I'm not suggesting that this is the only reason but I wonder how much of these changes is to do with legal liability issues. Now Freescale have allowed you to build security on top of theirs does this then absolve them. "Why didn't you build a good secure method on top of ours?" Also I suppose the tool makers could provide a lower level of security by providing a simple standard method that you could implement within your code and that they could then provide support for. This would give similar level of security than before. However these days being nice has legal ramifications.

As I showed in the thread you referenced backdoor access can be made as simple as pressing a button or as complicated as you like. Press a button send a byte over SCI wait between 3 and 4 seconds then send another etc etc.

Another issue is that before the device was secured by default and you could not forget the method or key for access. Idiot proof!!??
Now you need to deliberately secure the device, provide a key, then remember the key and the access method.

Oh, and to answer your specific question at the end.....
Yes you can use the backdoor facility to access a secure device and read all FLASH and RAM which of course includes the sections you have used for EEPROM emulation.

Message Edited by peg on 2007-03-1711:34 AM

Message Edited by peg on 2007-03-1706:46 PM

0 Kudos

REPLY

03-19-2007 04:45 PM • 633 Views

irob
Contributor V

Great expertise, Peg, as usual. Thanks for your reply!

It's beginning to sink in now. Let me see if I get this right: Due to the radically different architecture, a programmer such as P&E's can no longer directly unlock a part.

Instead, the designer would have to make some sort of custom box (such as an RS232 terminal link) and then program the target to unlock with the use of the

backdoor key under certain user conditions.

At least it's still possible to reveal portions of the flash.

0 Kudos

REPLY

03-20-2007 10:31 AM • 633 Views

Alban
Senior Contributor II

Hi,

To unlock the part, you need to write the backdoor key from within the software.
Therefore your software inside the MCU needs to have routines to prepare this.

To be more flexible during development you could assign that an IRQ would start the unsecure sequence.
For more security, it is advised not to store the key in the MCU and to use a combination like SCI/SPI/msCAN to send it to the MCU and then a kind of command to execute the unsecure firmware you need to have planned.

Cheers,
Alban.

0 Kudos

REPLY

03-20-2007 04:08 PM • 633 Views

bigmac
Specialist III

Hello,

Yet another possibility is to have the unlocking occur automatically (within the initialisation process) during the product development phase, and simply remove the unlocking code completely. once development is complete and the product is released.

For a product in the field, I wouldn't see any need for debugging capability using BDM, assuming the existing code would be replaced if an update were to occur.

Regards,
Mac

0 Kudos

REPLY

03-20-2007 04:21 PM • 633 Views

irob
Contributor V

Bigmac, except in the case of using the MCU's flash as EEPROM. When product comes back as RMA, for instance, it is helpful to examine its history. It would be helpful to unlock it, and view the saved user parameters.

0 Kudos

REPLY

03-20-2007 04:47 PM • 633 Views

bigmac
Specialist III

Hello Rob,

I assume that the contents of the flash EEPROM require a much lower level of security than the program code. So there

may be alternative means of dumping the data, other than via the BDM. Special SCI commands come to mind.

The additional code required to do this might be of comparable complexity to a "secret" mechanism for entering the back-door key, but would leave the program code more secure.

Regards,
Mac

03-20-2007 10:19 PM • 633 Views

peg
Senior Contributor IV

Hi,

Mac said:

Yet another possibility is to have the unlocking occur automatically (within the initialisation process) during the product development phase, and simply remove the unlocking code completely, once development is complete and the product is released.

Not sure at all on the purpose of this. This is locking the door and leaving the key in, then throw the key away.
How is this better than not locking the door then locking it never having had a key.

then said:
For a product in the field, I wouldn't see any need for debugging capability using BDM, assuming the existing code would be replaced if an update were to occur.

This bit I agree with and is what my thinking and is what I have done.

However, if I was using emulated EEPROM I think I would enable and implement backdoor access.
On products where I implement a serial port I always implement a serial command that reads out RAM.
This is very useful for some quick debugging with just a terminal. This sort of thing could easily be extended for the purposes of reading the "EEPROM" However...
Using backdoor access to read the "EEPROM" also allows checking failed units for memory corruption from failed/runaway "EEPROM" code and full code.
This is because I have had this happen in the past when in-application programming was done.

The difference in "complexity" to implement would depend on the complexity of the backdoor access. Backdoor could be simpler/smaller.