

6 Скрипт инициализации

В этой главе рассказывается о назначении скриптов инициализации и о том, как редактировать скрипты в H-Jtag и H-Flasher.

6.1 Назначение команд скриптов

H-Jtag имеет 6 команд скриптов: *Setmem*, *Getmem*, *Delay*, *SysReset*, *SoftReset* и *SetPPMR*. Назначение каждой команды описано в табл. 6-1. Пять команд из них могут быть скомбинированы для инициализации различных систем.

Табл. 6-1 Команды скриптов

Команда	Назначение
<i>Setmem</i>	Задаёт значение памяти
<i>Getmem</i>	Читает значение памяти
<i>Delay</i>	Добавляет задержку
<i>SysReset</i>	Выполняет сброс системы
<i>SoftReset</i>	Выполняет сброс программы
<i>SetPPMR</i>	Задаёт периферийный порт CP15 регистра перераспределения памяти в ARM11

Примечание: Сейчас можно задать только 6 команд скриптов. Но эти команды обслуживают большинство ситуаций. В будущем, набор команд может быть расширен для поддержки новых требований.

6.1.1 Setmem

Setmem – очень нужная скрипт команда. Эта команда может быть использована для задания значения памяти, которая имеет регистры распределения в памяти.

Формат скрипт команды следующий:

Setmem bit-width dest-address value

- *Setmem* – команда;
- *Bit-Width* – Ширина в битах операции записи, которая может быть 8, 16 и 32 бита;
- *Dest-Address* – Адрес назначения операции записи. Проследите за выравниванием этого адреса;
- *Value* – Величина значения, записываемого по адресу назначения.

Примеры:

Setmem 08-Bit 0x0 0x12 – Запись 0x12 по 0x0, ширина в битах - 8

Setmem 16-Bit 0x0 0x1234 – Запись 0x1234 по 0x0, ширина в битах - 16

Setmem 32-Bit 0x0 0x12345678 – Запись 0x12345678 по 0x0, ширина в битах - 32

6.1.2 Getmem

Эта команда может быть использована для чтения значения по заданному адресу памяти.

Формат скрипт команды следующий:

Getmem bit-width dest-address

- *Getmem* – команда;
- *Bit-Width* – Ширина в битах операции чтения, которая может быть 8, 16 и 32 бита;
- *Dest-Address* – Адрес назначения операции чтения. Проследите за выравниванием этого адреса;

Примеры:

Getmem 08-Bit 0x0 – Чтение значения по адресу 0x0, ширина в битах - 8

Getmem 16-Bit 0x0 – Чтение значения по адресу 0x0, ширина в битах 16

Getmem 32-Bit 0x0 – Чтение значения по адресу 0x0, ширина в битах 32

6.1.3 Задержка

Команда задержки используется для добавления определённой задержки между двумя скриптами. Некоторым операциям необходимо время, прежде чем появится эффект. Эта команда может быть использована как пустая операция, для ожидания, пока завершится предыдущий скрипт.

Формат скрипт команды следующий:

Delay time (millisecond)

- *Delay* – команда;
- *Time* – задержка в миллисекундах.

Примеры:

Delay 100 – Задержка 100 миллисекунд;

Delay 5000 – Задержка 5000 миллисекунд.

6.1.4 SysReset

SysReset – команда, выполняющая сброс системы. Эта команда может быть использована только как первая команда в скрипте пользователя. Если *SysReset* находится в другой позиции, она игнорируется.

Внимание: *SysReset* используется только как первая команда в скрипте пользователя.

Формат скрипт команды следующий:

SysReset (без параметра)

- *SysReset* – команда.

6.1.5 SoftReset

SoftReset – команда, выполняющая сброс программы. Основным назначением команды является сброс регистра контроля CP15. Для подключенных устройств с CACHE и MMU, эта команда может быть использована для запрета CACHE и MMU. Для этих типов устройств, программа или ОС внутри флеш памяти может конфигурировать MMU и выполнять запутанную операцию перераспределения памяти. Для перепрограммирования устройства, необходимо запретить CACHE и MMU с помощью *SoftReset*. После запрещения CACHE и MMU становится просто управлять схемой памяти и сделать ей такой, какой нужно.

Формат скрипт команды следующий:

SoftReset (без параметра)

- *SoftReset* – команда.

6.1.6 SetPPMR

SetPPMR – команда, используемая для задания периферийного порта CP15 регистра перераспределения памяти в ARM11.

Формат скрипт команды следующий:

SetPPMR Value

- *SetPPMR* – команда;
- *Value* – Новое значение для регистра PPMR.

Пример:

SetPPMR 0x70000013 – Задаёт регистру PPMR 0x70000013.

6.2 Редактирование скриптов инициализации

Оба средства H-Jtag и H-Flasher имеют редактор скриптов. Редакторы показаны на рис. 6-1 и рис. 6-2 соответственно. Вообще, редакторы H-Jtag и H-Flasher одинаковы. В введении к этой секции они объединены.

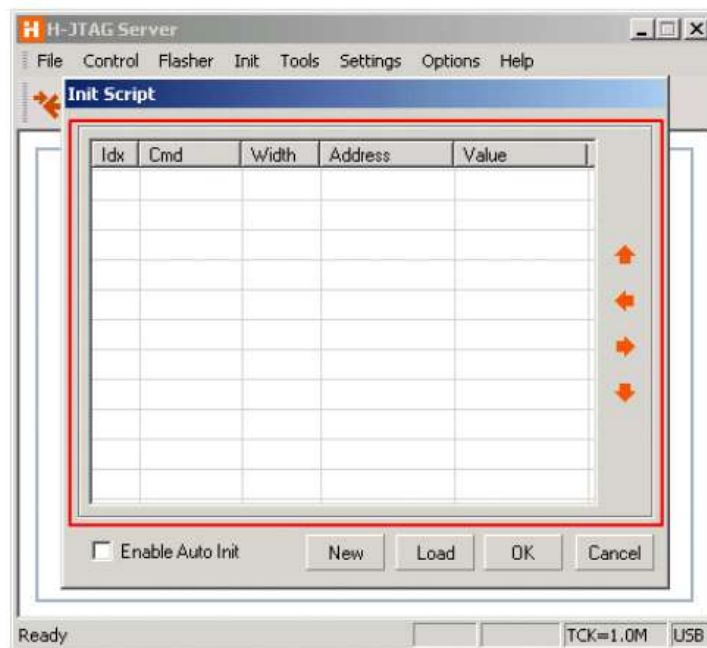


Fig 6-1 H-Jtag Script Editor

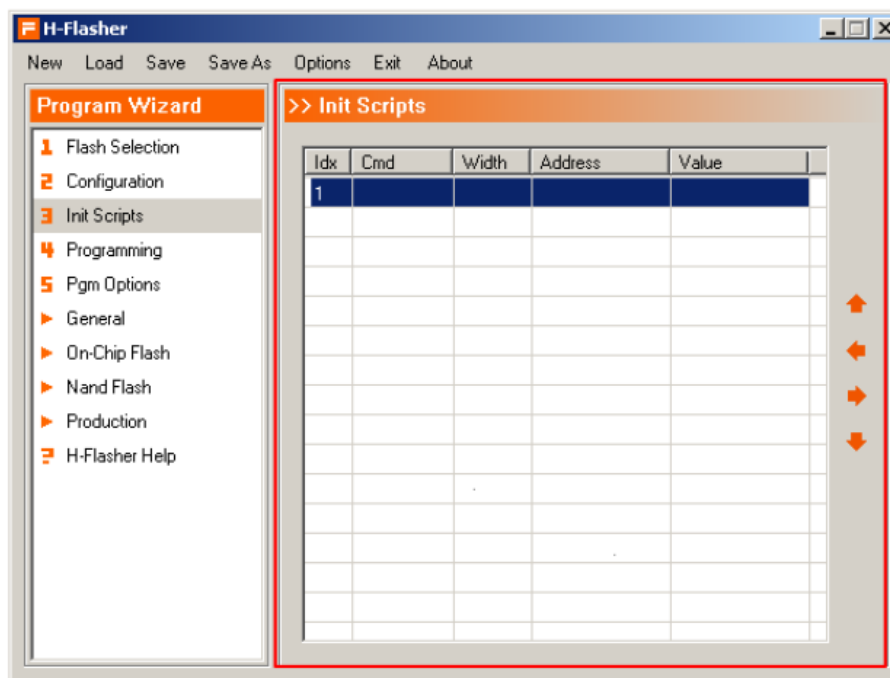






Fig 6-2 H-Flasher Script Editor

6.2.1 Кнопки редактирования

В редакторе скрипта четыре кнопки редактирования. Эти кнопки используются для добавления, удаления, перемещения вверх или вниз скрипта. Подробное описание приведено ниже

-  Перемещает скрипт вверх
-  Добавляет новый скрипт
-  Удаляет выбранный скрипт
-  Перемещает скрипт вниз

6.2.2 Редактирование нового скрипта

Для каждого нового скрипта, пользователю сначала нужно выбрать команду и, затем, задать для неё определённый параметр. Для добавления нового скрипта кликните "Add" справа. После того, как новый скрипт был добавлен, редакторы выглядят, как на рис. 6-3.

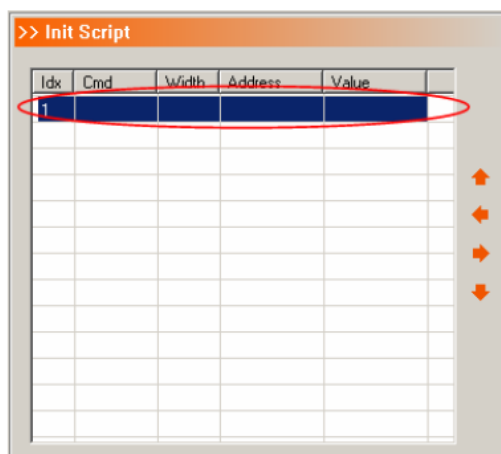


Fig 6-3 Add a New Script

Список команд можно увидеть, дважды кликнув на столбец **Cmd**. Список команд отображён на рис. 6-4. Пользователь выбирает нужную команду из списка.

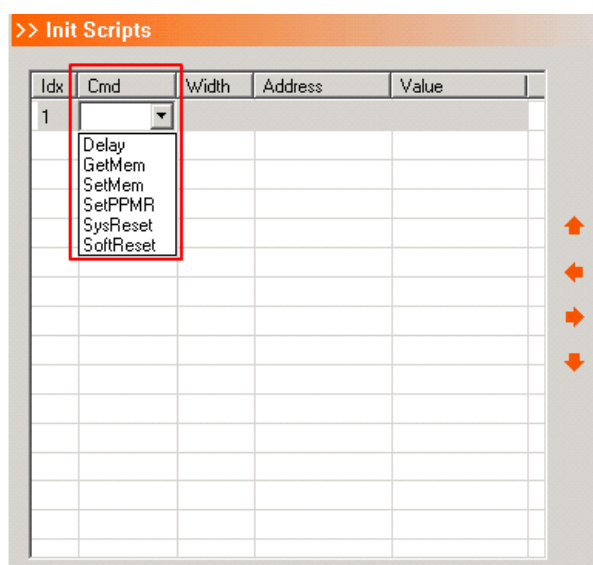


Fig 6-4 Command List

Если выбран *SysReset*, не требуется никаких параметров. На рис. 6-5 показано, как выглядит редактор после добавления скипта *SysReset*.

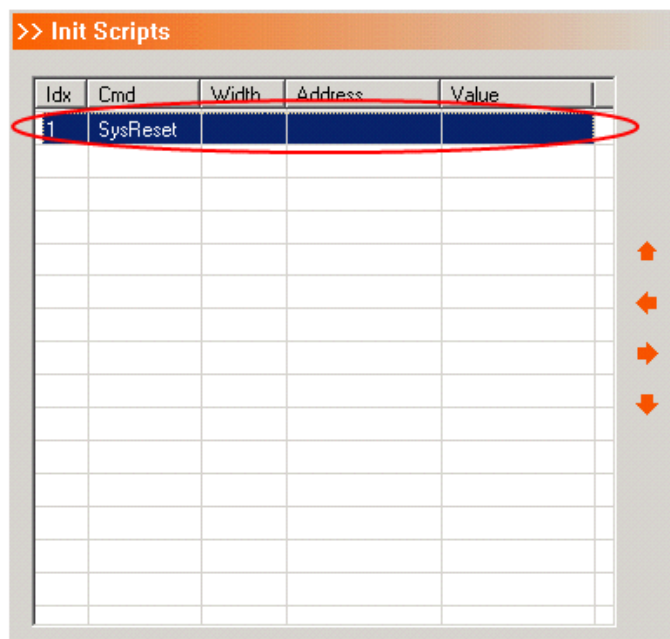


Fig 6-5 SysReset Script

Если выбран *SoftReset*, не требуется никаких параметров. На рис. 6-5 показано, как выглядит редактор после добавления скипта *SoftReset*.

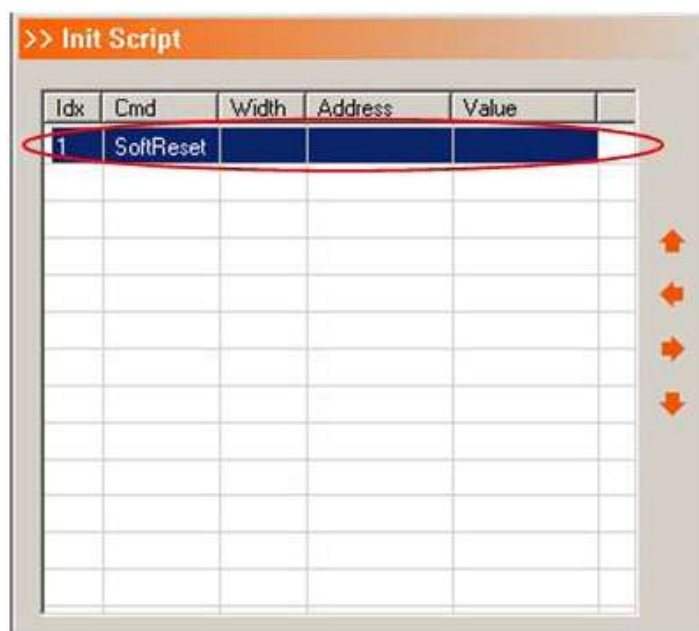


Fig 6-6 SoftReset Script

Если выбрана команда *Delay*, значение задержки должно быть задано. На рис. 6-7 показан скрипт с задержкой 5000 миллисекунд.

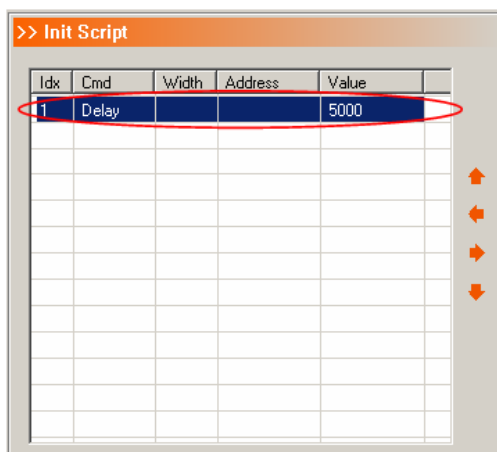


Fig 6-7 Delay Script

Если выбрана команда *Setmem*, всё должно быть задано: ширина в битах, адрес назначения и нужное значение. Когда выбрана команда *Setmem*, ширина в битах отображается после двойного клика в столбце **Width** (рис. 6-8). Пользователь сможет выбрать ширину в битах из списка. Далее, пользователю необходимо задать адрес назначения (столбец **Address**) и значение (столбец **Value**). На рис. 6-9 показан скрипт *Setmem 32-Bit 0x0 0x12345678*, который пишет значение 0x12345678 по адресу 0x0.

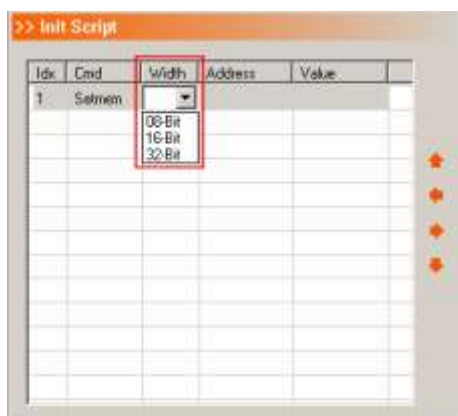


Fig 6-8 List of Bit-width



Fig 6-9 Setmen Script

Если выбрана команда *Getmem*, всё должно быть задано: ширина в битах и адрес назначения. Когда выбрана команда *Getmem*, ширина в битах отображается после двойного клика в столбце **Width** (рис. 6-10). Пользователь сможет выбрать ширину в битах из списка. Далее, пользователю необходимо задать адрес назначения (столбец **Address**). На рис. 6-11 показан скрипт *Getmem 8-Bit 0x10000000*, который читает значение 8-битное значение адресу *0x10000000*.

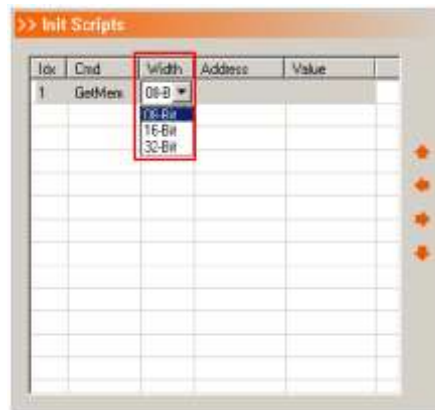


Fig 6-10 List of Bit-width

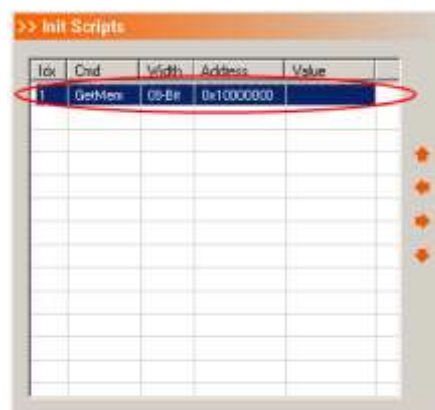


Fig 6-11 Getmem Script

Если выбрана команда *SetPPMR*, необходимо задать значения регистра PPMR. На рис. 6-12 показан скрипт, который задаёт регистру PPMR значение *0x70000013*.



6-12 SetPPMR Script

6.2.3 Добавление комментариев к скрипту

Для каждой команды скрипта пользователь может добавить комментарий. В редакторе скрипта, дважды кликните на индекс номера перед командой, появляется всплывающее окно, как показано на рис. 6-13. Пользователь в этом окне может увидеть существующий комментарий или добавить новый.



Fig 6-13 Script Comment