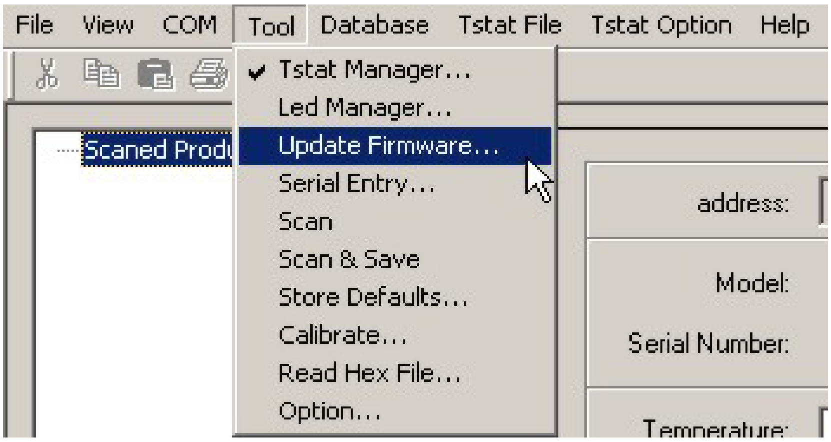
Instructions for Updating Devices with ISP

Nov.11

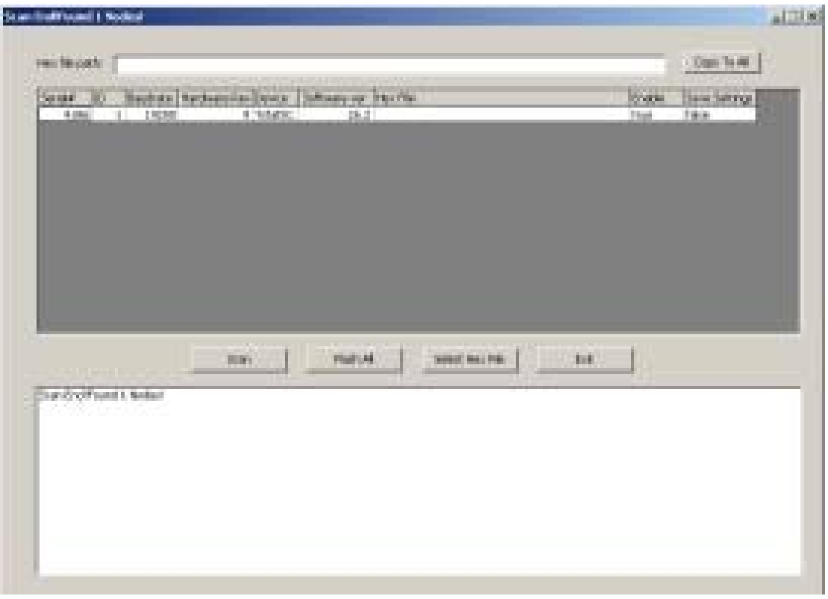
For AP devices tha utilize the AP ISP, the flash update must be done using the provided MPC manager.

To perform a firmware update, follow these instructions:

1. Download and install the MPC manger software: http://www.automatikprodukter.se/ftp/software/MPCSoftware.zip.
2. Connect the device to the serial port of your computer using RS232-485 converter included in the package.
3. Power up the device
4. Open the MPC Manager software and select Update Firmware from the Tool menu:



5. The software will now open the Update Firmware window and will scan for available devices



6. For each device that is found, you can specify the hex file to be used for the update.
   Do this by clicking in the HEX File column of the row you wish to specify.
   Alternatively you can click Select Hex File and the Copy to All if all devices are to receive the same file.
   You can also choose to save the current settings or to load the default settings by selecting True or False from the Save Settings column

7. At this point simply click Flash All and the software will update each device one by one

Protocol for Developers Wanting to Update Devices with ISP

All devices programmed with ISP are capable of being updated over the RS485 network.

The master on the network sends a command to a particular device, which forces it to go into a 'flash update mode'.

The device first resets itself and then jumps to the In System Programming (ISP) code section.

Note that all non-volatile parameters should be read and saved prior to this for safe keeping.

OBSERVERA: Multi-Write Command of the Modbus protocol is used

Protocol

In order for the front end to communicate with the ISP flash, a series of registers have been defined, which are used as control registers for the Update functions.

Reading and writing to these registers will allow the Front end to monitor the status of the update process.
They are stored in the non-volatile memory space to keep track of the steps attempted and completed.
Below is a description of these control status registers.

Register	Register address	Description
EEPROM_VERSION_NUMBER	4	Software Version
EEP_ADDRESS	6	ID number of the device
EEP_UPDATE_STATUS	16	Update Register state

Table 1. Flash Update Function Registers

It is important to note 'EEP_UPDATE_STATUS' which is located at register address 16.
Writing to this register will cause the device to either reset itself, erase its flash or start programming depending on the action being taken.
Below is a description of the values and explanation of the EEP_UPDATE_STATUS register.

Function	Value	Description of EEP_Update_Status
Update initialize	7Fh	Tell the MPC to reset and jump into the ISP to be in update mode
Update ready		MPC is in the ISP and ready to update
Erase flash	3Fh	Tell the MPC to erase Flash Memory
Erase done		Erase Flash Memory done
Start Programming	1Fh	Start Programming - In upload state
Normal State	01h	Update is complete, MPC reboots with new flash image

Table 2. EEP_UPDATE_STATUS register value description

- For the device to jump into update mode, a write command of value 7Fh must be sent to the EEP_UPDATE_STATUS.
  The device will then reset itself and will know to jump into ISP mode.
  Note: the device will not send any response in this step.
  To verify that the FBM is in ISP mode, the same write command must be sent again (write 7Fh to register #16), at which point the FBM will respond with a regular modbus response (7FH at regsiter16).
  This is necessary for clearing the Interrupt vectors and making sure all RAM memory is cleared.
- All Modbus communication commands are always followed by a response.

  This Flash Update Protocol makes use of that criteria and thus only sends a response once the action has been completed.
  Therefore the 'update initialize' and 'erase flash' step require a longer timeout period than the 'programming' step, (250ms and 500ms respectively)
- Sending a write command of value 1Fh to EEP_UPDATE_STATUS will force the device to erase its entire flash memory.

  Once the response is received, the device is ready to download the data of the new firmware.
- Sending a write command of value 1Fh to EEP_UPDATE_STATUS will let the device know it is about to receive new firmware.

  The device is now ready to accept the new hex file and will maintain a running tally of the current programming location in the EEP_UPDATE_PTR.
- At this point, the data must be sent using the multiple-write command.

  Packets can be of size 1 data byte to a maximum of **128 data bytes.**
- In the event of an interrupted flash update, the master can poll the EEP_UPDATE_PTR and begin programming from this location.

Example of a Programming Routine

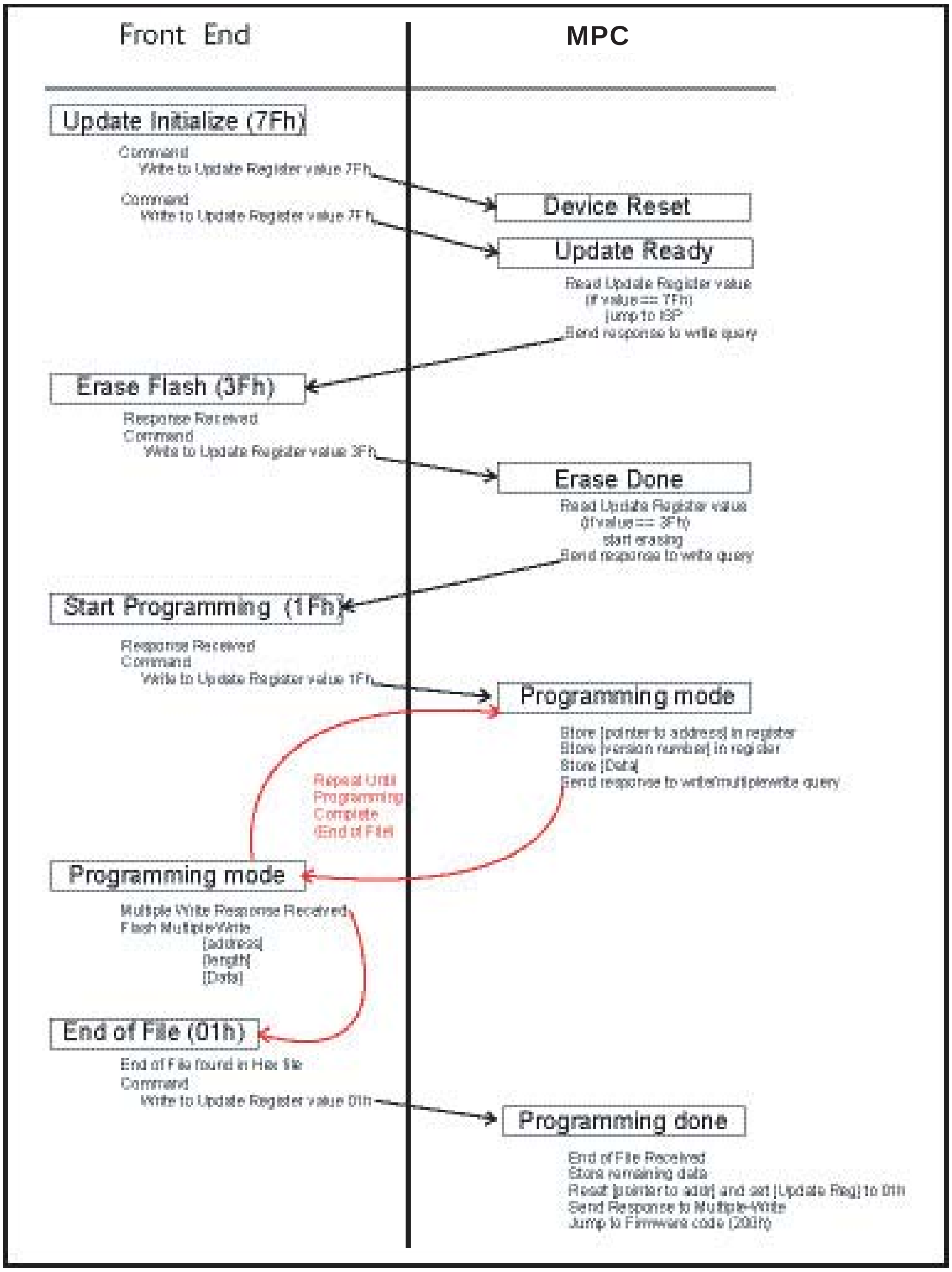
The ISP has been designed using polling vectors rather than Interrupt vectors in order to free up as many interrupts for the program itself.

Given that polling is now used, communications is more susceptible to timing and response delay problems.

Therefore, when sending a write function or multiple-write function to the ISP device, a short timeout delay is required before receiving a response (¨cirka 20ms).

If a response was not received during that period of time the FRONT END would need to resend the data once again.

Below is a diagram representation of the Flash-Update Protocol.


Front End
MPC
Update Initialize (7Fh)
Command
Write to Update Register value 7Fh
Command
Write to Update Register value 7Fh
Device Reset
Update Ready
Read Update Register value
(if value== 7Fh)
jump to ISP
Send response to write query
Erase Flash (3Fh)
Response Received
Command
Write to Update Register value 3Fh
Erase Done
Read Update Register value
(if value== 3Fh)
start erasing
Send response to write query
Start Programming (1Fh)
Response Received
Command
Write to Update Register value 1Fh
Programming mode
Store [pointer to address] in register
Store [version number] in register
Store [Data]
Send response to write/multiplewrite query
Repeat Until
Programming
Complete
(End of File)
Programming mode
Multiple Write Response Received
Flash Multiple-Write
[address]
[length]
[Data]
End of File (01h)
End of File found in Hex file
Command
Write to Update Register value 01h
Programming done
End of File Received
Store remaining data
Reset [pointer to addr] and set [Update Reg] to 01h
Send Response to Multiple-Write
Jump to Firmware code (200h)

Example of a Programming Routine (Front End Side)

Update initialise

1 - Send the Modbus Write command to adress Update_Register value 7Fh
The device will reset itself.
Make sure all volatile information be saved prior to this step
Device will not send a respond

2 - Send Modbus Write Command to address Update_Register value 7Fh again.
A response will be received if the Device has properly reset itself and booted under ISP mode.

Earse Flash

3 - Send the Modbus Write Command to adress Update_Register value 3Fh
A response will be received once the Device has properly set itself for programming mode
This will step require a longer response timout period (approx 500ms).

Tart Programming

4 - Send the Modbus Write Command to adress Update_Register value 1Fh
A response will be received once the Device has properly Erase all Flash memory
This will step require a longer response timout period (approx 500ms).

Programming Mode

5 - Extracting data from Inte Hex file.A typical line would look like the following:
: 10 0080 00 AF5F67F0 602703E0 322CFA92 007780C3 FD

6 - Verify checksum
10 + 00 + 08 + 00 +AF + ... + C3 + FD = 900
If two last digits of the sum is zero, Hex file is correct

7 - Send data using Modbus Multiple- Write command
Adress 0080h
Data length of 10h
Data AF5F67F0 602703E0 322CFA92 007780C3

8 - Repeat step 5 through 7 until end of Hex file is reached
IMPORTANT NOTE to ensure proper reset of the device, the value at address register 000h of the Goal chip must remain as FF.

Most (but not all) of AP:s Hex file will contain this line
:03 0000 00 020200 F9
Data written to the Goal Flash register MUST be modified from 020200 to FF0200

End of file

9 - End of file found in Hex file:
:00 0000 01 FF
Bit 7 and 8 are 01

10 - Send Modbus Write Command to address update_Register value 01h
This will cause the device to reset itself and boot in normal operation mode.

To Resume a Previously Interruped Programming Routine

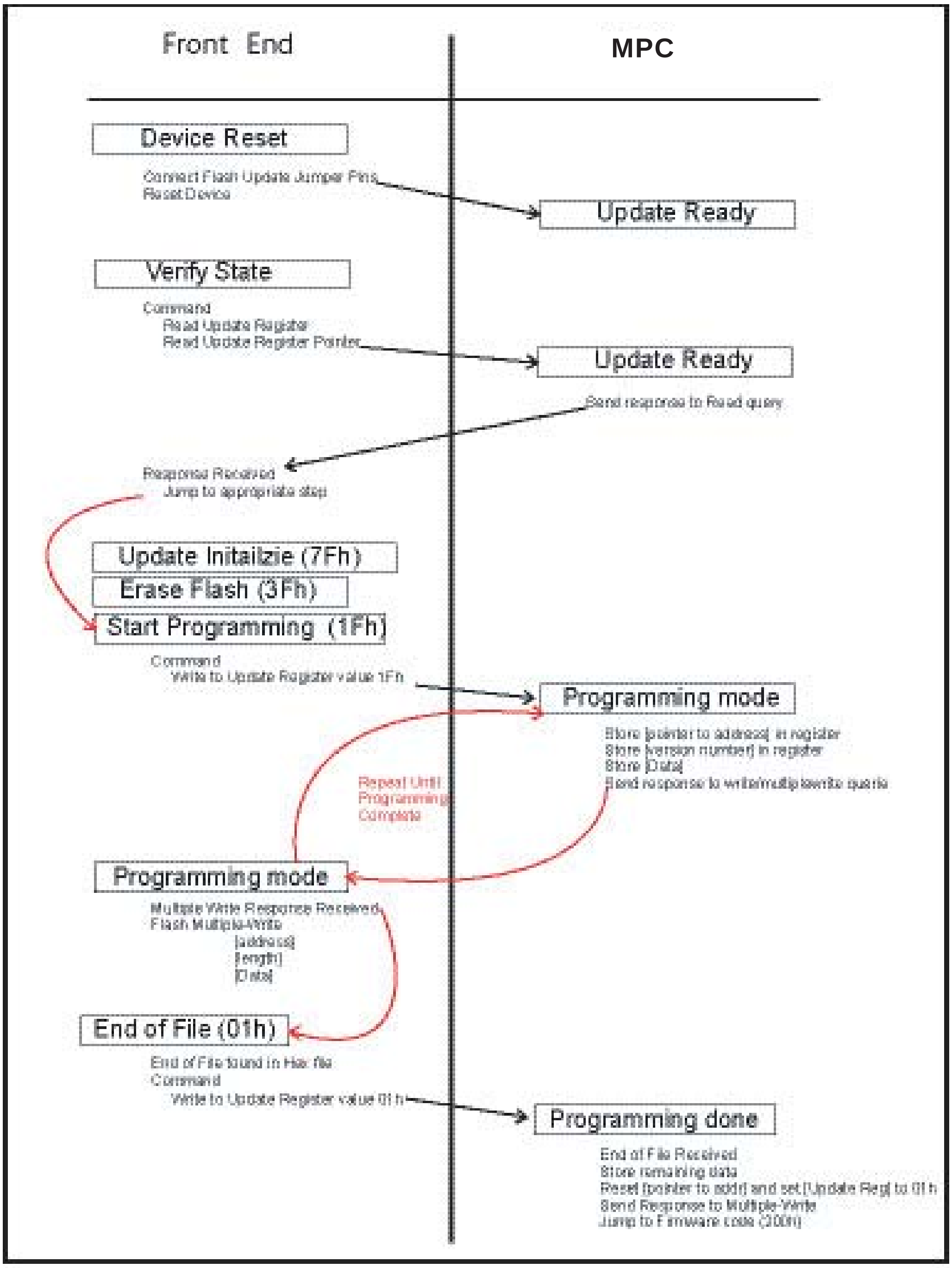
The EEP_UPDATE_STATUS register keeps track of which step is being performed during the update protocol and the EEP_UPDATE_PTR keeps track of which register is currently being written to.

- If the device was in the Erase Flash mode, the EEP_UPDATE_STATUS register will read 3Fh. The Front End is then required to repeat this step and follow up from there.

- If the device was in the Programming mode, the EEP_UPDATE_STATUS register will read 1Fh. The Front Ends then needs to read the EEP_UPDATE_PTR.

  <u>Thus, in order to resume this step the Front End needs to re-write to this register again</u> and then follow up from there.

The following diagram represents the update resume procedure.


Front End
MPC
Device Reset
Connect Flash Update Jumper Pins
Reset Device
Update Ready
Verify State
Command
Read Update Register
Read Update Register Pointer
Update Ready
Send response to Read query
Response Received
Jump to appropriate step
Update Initailzie (7Fh)
Erase Flash (3Fh)
Start Programming (1Fh)
Command
Write to Update Register value 1Fh
Programming mode
Store [pointer to address] in register
Store [version number] in register
Store [Data]
Send response to write/multiplewrite querie
Repeat Until
Programming
Complete
Programming mode
Multiple Write Response Received
Flash Multiple-Write
[address]
[length]
[Data]
End of File (01h)
End of File found in Hex file
Command
Write to Update Register value 01h
Programming done
End of File Received
Store remaining data
Reset [pointer to addr] and set [Update Reg] to 01h
Send Response to Multiple-Write
Jump to Firmware code (300h)

IMPORTANT:

In order for the device to jump into the ISP mode, it has to reset itself.

Upon reset, if the value at address register 0000h is FF the device will jump to the ISP code section.

This is a hardware criteria of the Goal Chip and an efficient way to jump to In System Programming mode while clearing all buffers.

The front end must ensure that only value FF is to be written to address register 0000h.

When reading the hex file, there will be a line such as this:

Data of the new Firmware		Modified data to be uploaded
;03 0000 00 020200 F9	need to change to this to	;03 0000 00 FF0200 FC
(Intel Hex format described below):		

Intel Hex File

All firmware files produced by our compilers are saved under the Intel Hex file format.

This format of record can be broken down in its different fields as described below.

Example of an Intel Hex file

Take for instance a typical message such as the following:

```
:ll  aaaa  tt  D1D2D3D4  D5D6D7D8  D9D0D1D2  D3D4D5D6  ee
:10 0080 00 AF5F67F0   602703E0   322CFA92   007780C3    61
```

- The first character (:) indicates the start of a record.
- The next two characters indicate the record length (10h).
- The next four characters give the load address (0080h).
- The next two characters indicate the record type. (00)
- Then we have our data
- The last two characters are a checksum (sum of all bytes + checksum = 00).

Record types:

- 00 - Data record
- 01 - End of file record
- 02 - Extended segment address record
- 03 - Start segment address record
- 04 - Extended linear address record
- 05 - Start linear address record

Intel Hex File

In the case where the tstat is locked, there is still a possibility to reboot the device and upload a new firmware.

This requires to physically link the jumpers of the Flash Update Jumper pins during restart:

- Power down the device
- Link the jumpers of the Flash Update Jumpers
- Power up the device

Doing the above steps will force the device to be in ISP mode so that new firmware can be loaded.

In order to return to normal operation once the upload has been done the Jumper needs to be removed and power need to be recycled.