Counting module PCD2/3.H100

Contents

1. Introduction

This document has two purposes. On the first hand it is the documentation of example programs for working with the H100 modules contained in the project PCD2-3_H100_Counting. These sample programs are intended to show how H100 modules can be used to count impulses or revolutions of incremental shaft encoders.

The following examples are covered in this document from chapter 2.3:

- Counting down from a preset value (0..65535) to 0 and setting CCO (Counter Controlled Output)
- Counting down from a preset value (>65535) to 0 and setting CCO

The second purpose is the documentation of the commands available for programming the H100 modules. A summary of these commands is present in chapter 3.

1.1 Functional description

The counting modules H100 for the PCD1, PCD2 and PCD3 are able to execute counting tasks (incrementing or decrementing) automatically and asynchronous to the PCD CPU. Further on the input filter of the counting modules is designed for higher frequencies of the input pulses (up to 20 kHz) than common digital inputs of I/O modules.

The H100 modules do feature functions that can be controlled by simple PCD instructions or a combination of them. Additionally the operating mode can be selected by a jumper on the module.

1.2 Possible application

- Counting revolutions or distances (impulses)
- Presetting a count value and switching off output CCO when the counter reaches 0 (or 65535, depending on the counting mode)
- Measuring signals counted only when particular conditions are met, e.g. a photoelectric barrier is covered
- Counting with recognition of count direction for incremental shaft encoders providing simple motion control

1.3 Hardware and software used

Hardware:

- PCD2.M480
- PCD2.E110 (for start signal)
- PCD2.H100 (for H100 examples)
- PCD8.K111 programming cable or USB cable)
- Some kind of pulse generator (can also be a switch which is manually operated). There is also a file called blinker that operates the output 16 which can be wired to the input A of the module. The use of this file and the output 16 are optional.

To use these examples with a controller type other than a PCD2.M480, the relevant type should be selected in the hardware settings for the project.
It is possible to load the configuration of the existing controller directly into the project, using the “Upload” function within the hardware settings (see chapter 2.1.2, Modification of hardware and software settings in PG5).

Minimum software version:

SAIA PG5 1.4.110 (the required FBs for the H modules are shipped and installed togehter with PG5)

2. Getting started with the H100

2.1 Preparing the PCD

The modules to be used could be inserted into any socket on the PCD2.M480. However, for using the examples without modifications the following configuration is required.

Slot 0 (Base address 0):	PCD2.E110
Slot 1 (Base address 16):	PCD2.A400 (optional)
Slot 2 (Base address 32):	PCD2.H100

In case another configuration is used the base addresses for the modules has to be entered in the program file for the module.
The modules are wired as per manual 26/737 “PCD1-PCD2_E14”.

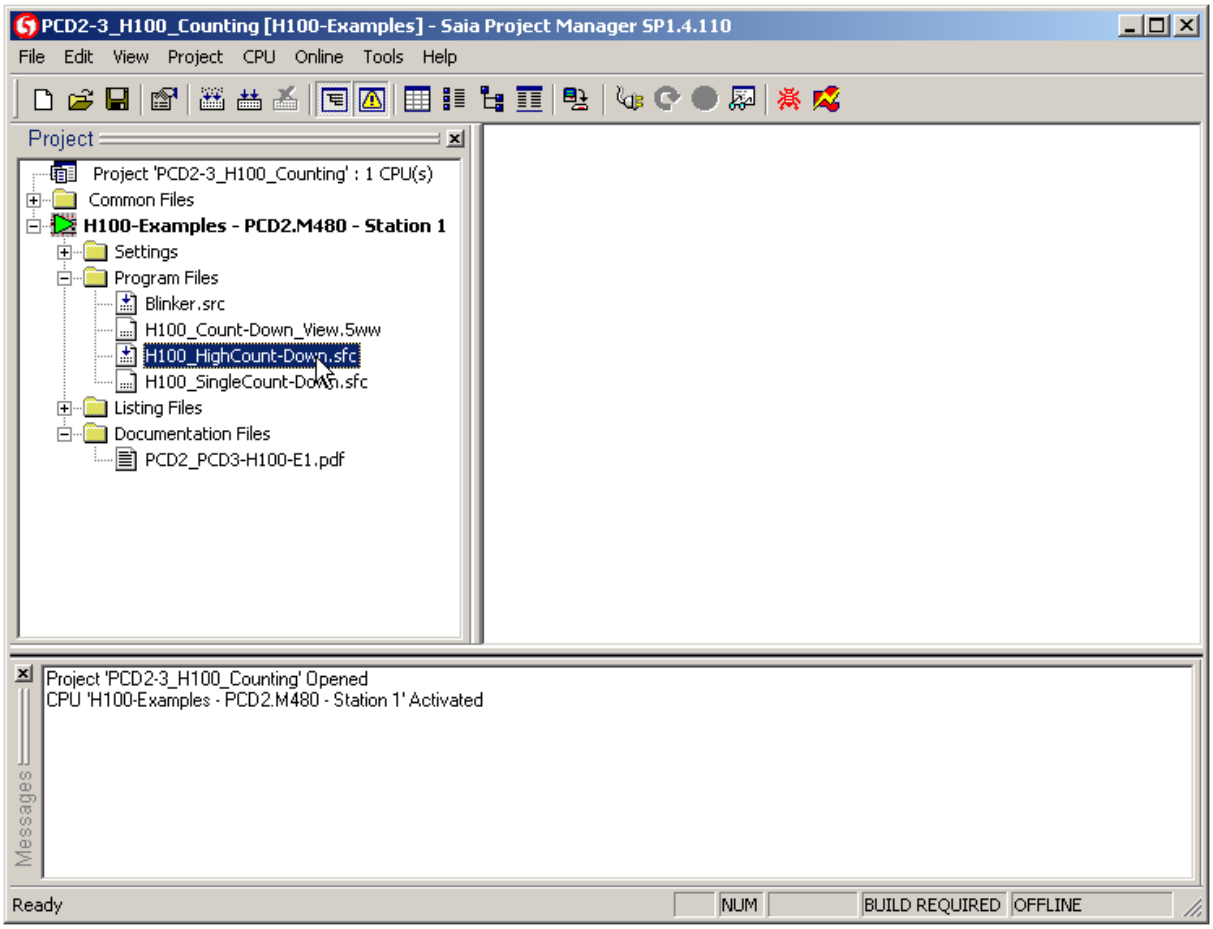
If the pulses are generated by the optional PCD2.A400 card in slot 1, the output 16 is to be connected to the input A on the PCD2.H100.

The I/O bus on the SAIA PCD Classic controllers is not designed for “hot plugging”. Before inserting or removing I/O modules, the controller must be disconnected.

2.1.1 Installation of the project

To install the project in your PG5 project directory, you should use the “Restore...” function from the “Project” menu in the PG5 1.4 Project Manager. This function will extract and copy the project into your project directory.

If you are using PG5 1.1, you will need to unzip the project and copy it into the PG5 project directory manually; in PG5 1.2 and 1.3 the “Restore…” function is located in the menu “File”.



This document can be found in the “Documents” folder in the project tree for the PG5 Project Manager, and can be opened directly from there by double-clicking on it.

2.1.2 Modifications of hardware and software settings in PG5.

The first step is to launch PG5 Project Manager (PG5 SPM).
The following procedure should then be followed:

- Connect the PCD to the PC with a PGU programming cable (PCD8.K111) or USB cable, and switch it on.
- Open the “Online settings” screen (in the Settings folder in the project tree of the Project Manager) and adjust it to either PGU or S-Bus USB, depending on your connection.
- Open the “Hardware settings” screen (in the Settings folder in the project tree for the Project Manager).
- Select “Upload” to load the hardware configuration for the PCD onto the PC. Then save by clicking “OK”.
- Open the “Software settings” screen (in the Settings folder in the project tree for the Project Manager).
- Set the dynamic range for the resources by clicking on “Set Default”, and confirm by clicking “OK”.
- Link the program file to be used (e.g. H100_SingleCount-Down.sfc). Right-click on the desired program in the project tree” and select “Linked”.

2.1.3 Modifying the code for the current PCD system

To ensure that the correct module is read and that the user units are correctly configured, the following settings should be entered:

- The base address of the socket in which the module is located must be defined (H100.BaseAddress). The base address is a multiple of 16 and can be read directly from the motherboard on a PCD1 or PCD2.
  On a PCD3, only the sockets are numbered (starting from 0). Here, the base address of the I/O bus can be calculated by multiplying by 16 (module base address = 16 * socket address).

Note that the socket address on expansion housings will be a continuation of the addresses from the previous housing.

2.1.4 Building and loading the project into the PCD

After making the adjustments explained in the preceding section, the project can be loaded into the controller following a “Rebuild All” (“CPU” menu, “Rebuild All...” option, or Alt+F2).

If the controller is already in a “Run” state, the system will ask whether the controller can be stopped. This will be the case during testing. The message is displayed for safety reasons, as it may not be permissible to stop the controller while in use on an existing installation.

2.2 Viewing values online

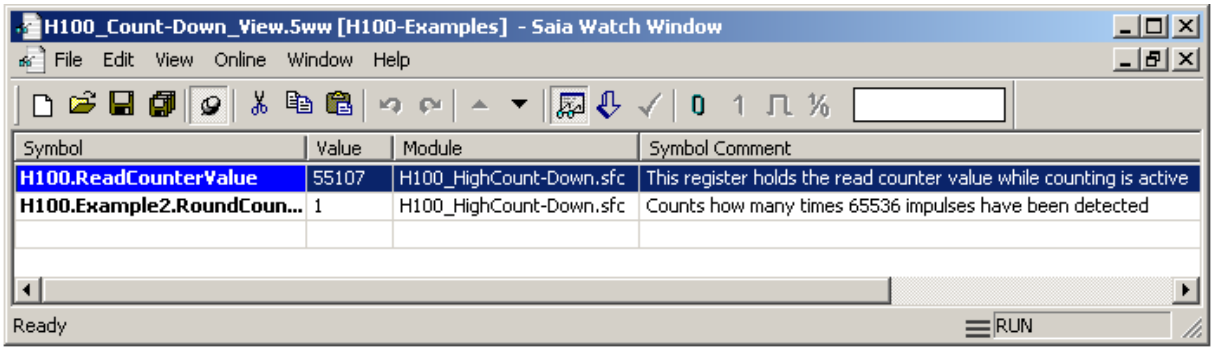
Once the program is loaded into the controller, an online connection can be established to the controller, in order to view the values online.
Clicking on the "Online" button connects the PC to the PCD. If the PCD is not yet in a Run state, it can be started with the curved green arrow on the toolbar.

The Watch Window (see next section) can now be used to view the values online.

Theoretically, it would also be possible to view the code online in the IL Editor. However, as the H modules may react to any access to the I/O bus, this should be avoided.

2.2.1 The SAIA Watch Window

To display and change the values from media on a screen, the Watch Window can be used. This can be opened by double clicking the file H100_Count-Down_View.5ww in the PG5 Project Manager tree.
The symbols to be displayed can be “dragged and dropped” into the window. For going online, hit the key “F9”.



2.3 H100 Examples

Counting tasks are always carried out in sequential programming:

1. Definition of the counting task
2. Waiting for the end of counting
3. Evaluation of the counting (e.g. time elapsed since start)

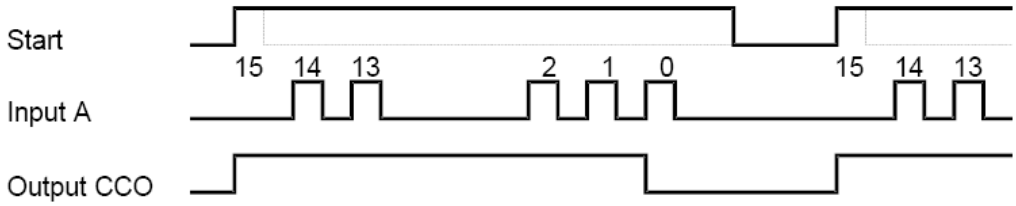
A logical conclusion is to write the user program in Graftec.

As the examples in the project PCD2-3_H100_Counting always use the same PCD2.H100 module on the PCD, only one program file can be run at a time. This means that either the file H100_SingleCount-Down.sfc or the H100_HighCount-Down.sfc can be linked. It is not possible compiling both at the same time!
The file Blinker.src can always be linked, even if there is no module inserted in slot 1 on the PCD.

2.3.1 Example 1: H100_SingleCount-Down.sfc

This example fulfils the following requests:

- When PCD input 0 receives a start signal, the counter should be loaded with the value 15 and the CCO output shall be switched on.
- After 15 count impulses have reached counter input A, the CCO output shall be reset.
- If the start signal is removed and then restored (after the first measurement has finished), a new sequence shall be started.
- The current count value shall always be present for visualisation (with e.g. the Watch Window H100_Count-Down_View.5ww).



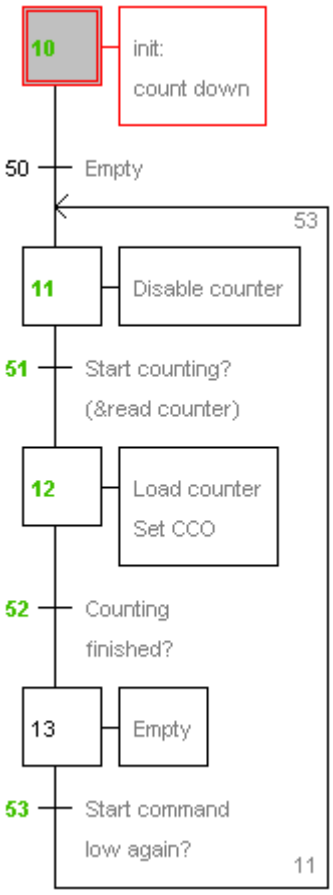
Procedure

- Single count (SC) count mode is selected --> Jumper in position "SC".
- The counter is loaded with the value 15.
- Down-counting to 0 takes place.

Input “A” is to be connected to the impulse source (either a manual switch, a pulse generator or the first output on the PCD2.A400 placed in slot 1).
Input "B" should be connected to 24V, so that count signals reach the counter (AND gate).

Note that on input A and B the input filter is designed for frequencies up to 20 kHz. As result, these inputs are sensitive on chattering signals.

Description of the single steps and transitions



SB 0: Sequential Block.
In Graftec the SB serves as a wrapper. Each self contained Graftec branch is contained in an SB.

IST 10: In the present structure, the IST is only processed at the first call of SB 0. Down-counting has been selected here.

ST 11: Stop the counter.
This command is only effective at the second pass of the SB, since the counter is not even active when it is first executed.
Stopping the counting is not absolutely necessary for the function of this example.

TR 51: The main task of this TR is to record the Graftec switching condition. This is when input "Start" is high.
In order for counter value to be recorded in each phase of the program, counter value should be read and, in this case, written to register H100.ReadCounterValue, before the switching condition is polled. This register can be viewed using the Watch Window.
If at the end of the TR the ACCU is high, both the TR and with it the switching condition are fulfilled and the next ST is processed. If at the end of the TR the ACCU is low, i.e. switching condition not fulfilled, the program returns to the calling COB and continues from there.
The next time SB 0 is called, ALL of the TR which was previously unfulfilled is executed again, thereby also refreshing the counter value.

ST 12: The counter is loaded with 15 via the H100.HelpRegister. The Output CCO is then set and the counter is started.
The last instruction line

```
RES         H100.BaseAddress+14
```

instructs the module to reset output CCO the next (first) time the counter flag is low.

TR 52: The counting module is now working independently. As soon as the preset number of signals has been received, the counter flag and output CCO are reset by the module itself.
For the program to continue to run, a switching condition must be defined. This is the logical state of the counter flag, which can be polled with

```
STL         H100.BaseAddress+0
```

Here too the register H100.ReadCounterValue is first refreshed with the current counter value.

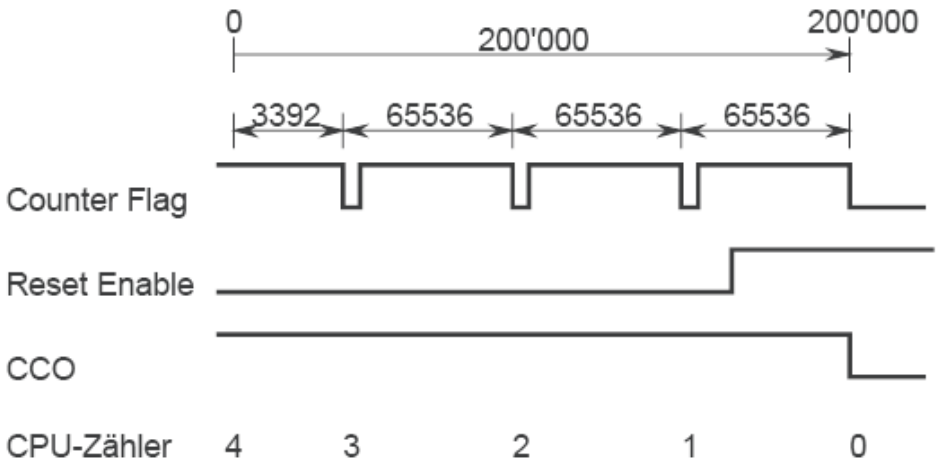
TR 53: The switching condition is for input "Start" to be low. The register H100.HelpRegister is refreshed with the current counter value.

Return to ST 11 where the counter is blocked.

2.3.2 Example 2: H100_SingleCount-Down_extended.sfc

The second example does work very similar to the first example but with the difference that more than 65535 impulses are to be counted before the CCO is reset.

This can be achieved by setting the CCO by the user program and only resetting it after several “rounds” in which the counter value has reached 0 (and thereby reset the counter flag). The rounds are counted in the Graftec structure by the PCD counter H100.Example2.RoundCounter.
After the counter flag has reached 0 the fourth time, the “Reset Enable” on the module is set by the user program (as result the physical CCO will be reset as soon as the counter reaches 0):



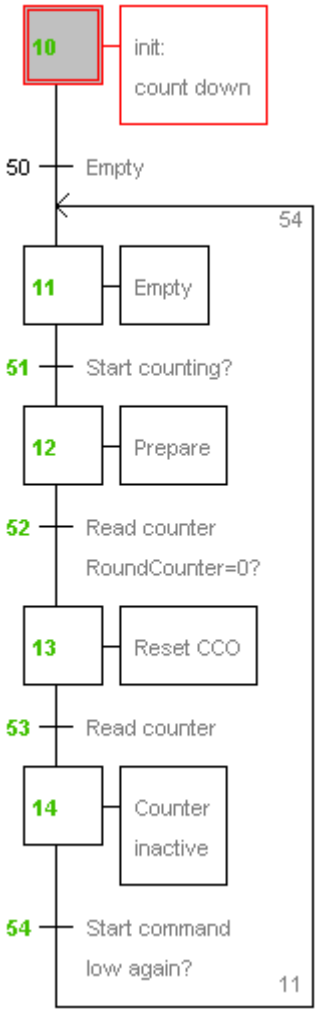
Procedure

- Single count (SC) count mode is selected --> Jumper in position "SC".
- At the beginning the counter is loaded with the value 3392 and the CPU counter (H100.Example2.RoundCounter) is set to 4.
- Down-counting to 0 takes place. Each time the counter flag is set to 0, it is reset to 1 and the H100.Example2.RoundCounter is decremented
- As soon as the H100.Example2.RoundCounter is 0, the Reset Enable is set (causing the CCO to be set to 0 the next time the counter value reaches 0).

Input “A” is to be connected to the impulse source (either a manual switch, a pulse generator or the first output on the PCD2.A400 placed in slot 1).
Input "B" should be connected to 24V, so that count signals reach the counter (AND gate).

Note that on input A and B the input filter is designed for frequencies up to 20 kHz. As result, these inputs are sensitive on chattering signals.

Description of the single steps and transitions



IST 10: In the present structure, the IST is only processed at the first call of SB 0. Down-counting is selected here.

ST 11: Empty, only for Graftec structure purpose.

TR 51: The main task of this TR is to record the Graftec switching condition. This is when input "Start" (Input 0) is high.

ST 12: The counter is loaded with 3392 via the H100.HelpRegister. Additionally the H100.Example2.RoundCounter is loaded with 4. The Output CCO is then set and the counter is started.

TR 52: The counting module is now working independently. As soon as the preset number of signals has been received, the counter flag is reset by the module itself. This will be detected by the user program and as action each time the counter flag is set again and the H100.Example2.RoundCounter is decremented. The register H100.ReadCounterValueis refreshed with the current counter value.
The condition to switch to the next step is the RoundCounter being 0.

ST 13: The “Enable Reset” is set. As result, the output CCO on the module is reset the next time the counter of the H100 reaches 0.

TR 53 The register H100.ReadCounterValue is refreshed with the current counter value. The condition to switch to the next step is the counter flag = 0.

ST 14: The counter is deactivated here.

TR 54: Waiting until the start command (Input 0) is reset.

Return to ST 11.

.

3. Commands for programming the H100

The following functions for programming the H100 are available:

Write commands

- Select direction of count (SC mode only)
- Start/stop count
- Set CCO output
- Set counter flag
- Reset enable CCO output
- Load counter with a value between 0 and 65 535

Read commands

- Poll counter flag
- Read current counter value

The functions are controlled by single PCD instructions (SET/RES) or by combinations of instructions. Instructions containing a hardware address in the operand should be offsets from the module's base address (BA).

3.1.1 Write commands

- **Select direction of count (jumper in position "SC")**

```
SET       O BA+13   ; down-counting
RES       O BA+13   ; up-counting
```

- **Start / stop count**

```
SET       O BA+10   ; start counter
RES       O BA+10   ; stop counter
```

- **Set CCO output**

```
SET       O BA+14   ; direct setting of CCO output
RES       O BA+14   ; the CCO output will switch back
                    ; at the next counter flag reset.
                    ; (Reset Enable)
```

- **Set counter flag**

```
STL       O BA+0    ; Poll counter lag and
SET       O BA+11   ; again set counter flag
RES       O BA+11   ; when counter reaches zero.
(DEC      C y)      ; e.g. decrementing a PCD counter
```

- **Load counter with a value between 0 and 65 535**

```
LD          R x         ; PCD register R 0-4095
            Value       ; 0 ... 65’535
BITO        8           ; Load lower 8 bit
            R x
            O BA+0
SET         O BA+8      ; Execute function
RES         O BA+8
ROTR        R x
            8
ACC         H
BITO        8           ; Load upper 8 bit
            R x
            O BA+0
SET         O BA+9      ; Execute function
RES         O BA+9
```

When this load routine is executed, the counter flag is also automatically set high..

3.1.2 Read commands

- **Poll counter flag**

```
STL         O BA+0      ; ACCU becomes high (= H),
                        ; when counter flag is low (= L)
```

- **Read current counter value *)**

```
SET         O BA+12     ; For the refreshed display of
BITI        16          ; current counter value,
            O BA+0      ; this read routine must be
            R z         ; processed continuously,
RES         O BA+12     ; or e.g. once a second.
```

*) Read during counting can only be used for display purposes and **not** for compare functions.