# FANUC

# UNIVERSAL DEVICE DRIVER

This device driver enables you to create your own set of items from the Fanuc FOCAS library, without the need to write a single line of code! In addition to monitoring and output a broad spectrum of process data it allows you to dynamically define and change items on a per machine basis with an INI file while CNCnetPDM is running.

The following functions are included and can be used in the standard version of this driver. If you need additional functions we can add them at a fixed price. Please contact us to get a quote!

- Read status information (cnc_statinfo)
- Read alarm status, program number in execution, program number of main program, actual sequence number, actual feed rate, actual spindle speed (via cnc_rddynamic2 or cnc_rddynamic)
- Read absolute, machine and relative position of axis and amount of distance to go (via cnc_rddynamic2 or cnc_rddynamic)
- Read all system and user parameters (via cnc_rdparam)
- Read macro values (via cnc_rdmacro)
- Read modal data from controller (via cnc_modal)
- Read alarm numbers and text (via cnc_alarm)
- Read PMC (PLC) data from all areas (via pmc_rdpmcrng)
- Read OEE machine state and execution mode (text and/or numeric)

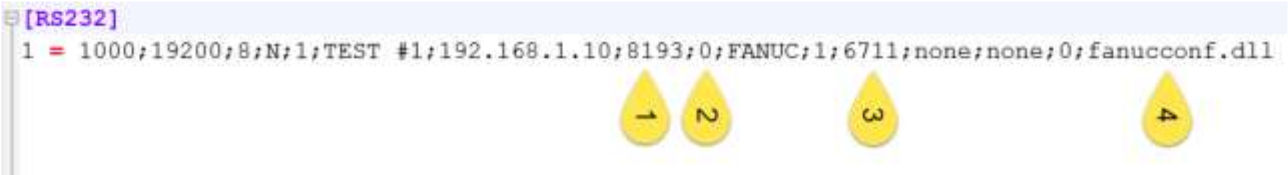
## LICENSING

This device driver requires the most recent version of CNCnetPDM and also works with a free license. However, in this mode you only get output for the first item configured in your INI file. With a valid license you are able to output the result of up to 30 functions per reading cycle, see licensing for details.

## SETUP

- Make sure that you have a connection from your PC to the controller and get data, use the fanucdata utility to do so. In case you use HSSB instead of Ethernet please read the additional instructions.
- If not already done download and install CNCnetPDM, if you don’t have time to study the user manual read at least the quick start guide.
- Alternatively you can also use our device driver test tool to work with the driver, no need to setup CNCnetPDM.ini in this case.

- Download the Universal Fanuc driver, if you do not have any Fanuc driver installed extract all content of fanucconf_dll.zip into the folder where you have CNCnetPDM installed. Otherwise it's sufficient to just extract fanucconf.dll to this folder.
- Edit CNCnetPDM.ini and add a new device as described in the quick start guide point 6. If you already have an entry in section [RS232] for a machine with a Fanuc controller you can simply copy and paste this line and change its content.



```
[RS232]
1 = 1000;19200;8;N;1;TEST #1;192.168.1.10;8193;0;FANUC;1;6711;none;none;0;fanucconf.dll
```

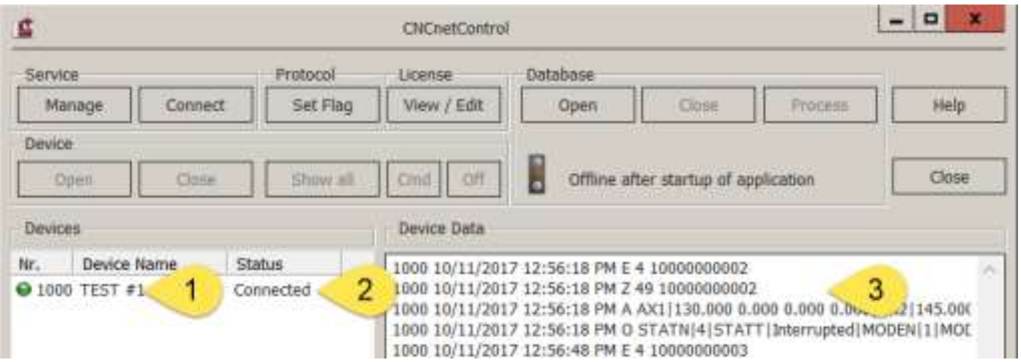
FIG 1: Add a Fanuc device

For Fanuc controllers the following 4 parameters are important:

1. The parameter (1) is the port you are using for communication with your controller, 8193 would be Fanuc FOCAS standard.
2. The next parameter (2), here 0 tells the driver what connection you are using, change it to 1 if you use 'real' HSSB or 2 if you use the HSSB functions of NCGUIDE. Zero and every higher number than 2 is Ethernet. In case of HSSB use 127.0.0.1 as IP Address and localhost as DNS Hostname.
3. Parameter (3) tells the driver how you'd like to acquire part counter values. If you enter 6711 the driver reads parameter 6711 that's standard on most controllers. Any other number means: Read the number as address from the (PLC) PMC (Area 9).
4. Change parameter above (4) to fanucconf.dll.
5. Please also set the following entries in section [GENERAL] of CNCnetPDM.ini to 1 to get all possible items:
   CollectStates = 1
   CollectCounters = 1
   CollectFeeder = 1
   CollectQuality = 1
   CollectOrders = 1

# USAGE

- Start CNCnetPDM, foreground program is sufficient (Start thread)
- CNCnetPDM automatically copies the original fanucconf.dll and appends the machine number as configured in the INI file, e.g. fanucconf_1000.dll for machine 1000.
- In addition an INI file with the same name is created by the device driver, e.g. fanucconf_1000.ini for machine number 1000.
- Double click CNCnetControl, if your device number is 1000 and the device name is TEST #1 the output should be similar to the one below:



Offline after startup of application

| Nr. | Device Name | Status |
|---|---|---|
| 1000 | TEST #1 | Connected |

```
1000 10/11/2017 12:56:18 PM E 4 10000000002
1000 10/11/2017 12:56:18 PM Z 49 10000000002
1000 10/11/2017 12:56:18 PM A AX1|130.000 0.000 0.000 0.00[illegible]2|145.00(
1000 10/11/2017 12:56:18 PM O STATN|4|STATT|Interrupted|MODEN|1|MOI
1000 10/11/2017 12:56:48 PM E 4 10000000003
```

FIG 2: CNCnetControl (Fanuc)

The machine (1) shows up as connected (2) which is good. On the right side (3) you see the acquired data:

- E = Machine state 4 would be interrupted mode, changes to 2 if the machine is in automatic mode and runs a program. 3 would be manual, 5 error.
- Z = Part counter value. If you're counting in cumulative mode you see the part counter value as shown at the operator panel.
- The lines starting with O, A or F contain the items you added via the INI file. If you didn't change the automatically created INI file you get something like the following:

1000 10/11/2017 12:57:18 PM O STATN|4|STATT|Interrupted|MODEN|1|MODET|Memory|MOTIO|0|PRGCU|4|PRGMN|4|SEQCU|11|FEEDR|0|SPSPD|1001|TOOL1|1|CUTFD|5000.000|ALMNR|0|ALMTX|No alarm|TOOL2|1.00000000|PMC01|255| 10000000004

Every item has a description followed by its value. All descriptions and values are delimited by pipe '|' symbols. This allows the most recent version of CNCnetPDM to create a database record for every item.

With an unmodified INI file these items are:

| ID | NAME | DESCRIPTION |
|---|---|---|
| 2 | STATN | OEE Machine state number |
| 3 | STATT | OEE Machine state text |
| 4 | MODEN | Controller mode number |
| 5 | MODET | Controller mode text |
| 6 | MOTIO | Motion of axis |
| 7 | PRGCU | Current NC program |
| 8 | PRGMN | Main NC program |
| 9 | SEQCU | Current sequence |
| 10 | FEEDR | Actual Feed rate |
| 11 | SPSPD | Actual Spindle speed |
| 12 | AX | Axis position + axis number (multiple values) |
| 11 | ALMST | Alarm status number |
| 13 | TOOL1 | Tool id (modal) |
| 14 | CUTFD | Cutting feed rate (parameter) |
| 15 | ALMNR | Alarm number |
| 16 | ALMTX | Alarm text |
| 17 | TOOL2 | Tool ID (system macro) |
| 18 | PMC01 | PMC value (area 0 address 118) |

FIG 3: IDs, names and description of default items

- The line starting with A contains the current axis position values for up to 7 axes. With an unmodified INI file and a machine with 2 axes you get something similar to the following:

```
1000 10/11/2017 01:15:48 PM A AX1|130.000 0.000 0.000 0.000|AX2|145.000 0.000 0.000 0.000|
10000000041
```

The axis number is appended to the name. The values for each axis are:

| | |
|---|---|
| First value: | Absolute position |
| Second value: | Relative position |
| Third value: | Machine position |
| Fourth value: | Distance to go |

# INI FILE DESCRIPTION

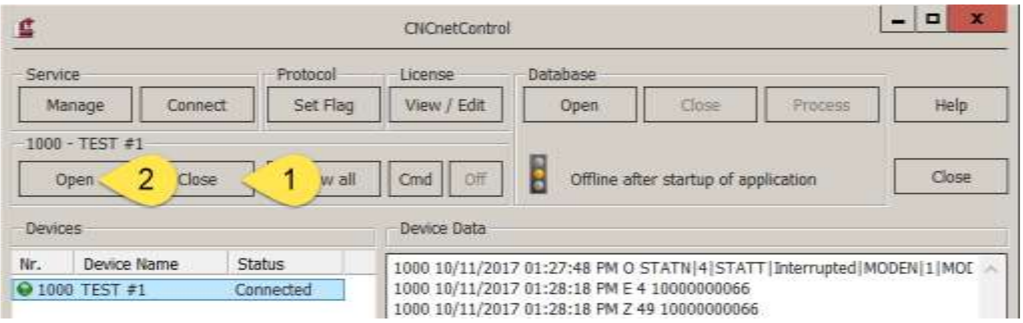
The device driver enables you to dynamically select items from the Fanuc FOCAS library, group, change, enable or disable them. The INI file automatically created by the device driver for every machine contains a section with the ID number of each item (see table above) e.g. [18] for PMC data:

```
[18]
Active = 1
Name = PMC01
Comment = PMC (area 0 address 118)
Command = pmc_rdpmcrng
Input parameter 1 = 0
Input parameter 2 = 118
Input parameter 3 = -1
Input parameter 4 = -1
Output item = idata
Output section = 1
```

You can switch acquisition of specific items on or off by altering the value of Active (0 = off, 1 = on). You can also adjust the name of the item by changing the value of Name, up to five characters are possible. For example if you do not need to acquire tool ID via the modal and the parameter function you can switch one of them off. If you do not need axis positions you can also switch it off by setting Active to 0 in section [12] of the INI file.

It is also possible to force output of items to section 1,2 or 3. Items in section 1 go to group prefixed with ‘O’, 2 = ‘A’ and 3 = ‘F’. Make sure you have entries CollectOrders, CollectFeeder and CollectQuality enabled (=1) in CNCnetPDM.ini.

To add or adjust items while CNCnetPDM is running open the INI file with a text editor such as notepad, make the desired changes and save the file. To apply the changes immediately you can click on the machine in CNCnetControl on the left side followed by clicking buttons ‘Close’ (1) and ‘Open’ (2) above the section ‘Devices’.



| Nr. | Device Name | Status |
|---|---|---|
| 1000 | TEST #1 | Connected |

```
1000 10/11/2017 01:27:48 PM O STATN|4|STATT|Interrupted|MODEN|1|MOD
1000 10/11/2017 01:28:18 PM E 4 10000000066
1000 10/11/2017 01:28:18 PM Z 49 10000000066
```

FIG 4: Reload device INI file

# MODIFY ITEMS

As stated the driver allows you to directly add various functions from the Fanuc FOCAS library. Although many of these functions are pretty complex we tried to make it as simple as possible.

To add a function from the library the following items in each INI file section are important:

Command = pmc_rdpmcrng

Input parameter 1 = 0
Input parameter 2 = 118
Input parameter 3 = -1
Input parameter 4 = -1

Output item = idata

Command is the name of the function (all lowercase). You can find it in the field with cyan background in the documentation right of WINAPI. For pmc_rdpmcrng this would be:

```
#include "fwlib32.h"


FWLIBAPI short WINAPI pmc_rdpmcrng(unsigned short FlibHndl, short
adr_type, short data_type, short s_number, short e_number, short length,
IODBPMC *buf);
```

FIG 5: Command and parameters

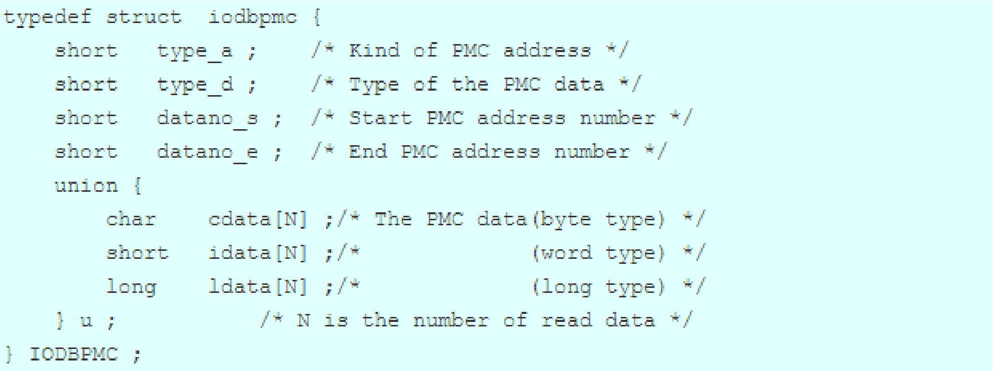
Input parameters 1-4 are the arguments after ‘unsigned short FlibHndl’ in the specified order. In this example

short adr_type = Input parameter 1, here 0 = G (Signal to PMC->CNC)

short s_number and short e_number = Input parameter 2 (as the driver queries one address per section, you can have multiple sections that call pmc_rdpmcrng!)

short data_type and short length are automatically calculated by the driver depending on the Output item you set in the INI file, here Output item = idata.

Many of the Fanuc FOCAS functions are able to output different items or data formats. For pmc_rdpmcrng you can find them in the field with cyan background that describes the structure, here iodbpmc.



```
typedef struct  iodbpmc {
    short   type_a ;    /* Kind of PMC address */
    short   type_d ;    /* Type of the PMC data */
    short   datano_s ;  /* Start PMC address number */
    short   datano_e ;  /* End PMC address number */
    union {
        char    cdata[N] ;/* The PMC data(byte type) */
        short   idata[N] ;/*             (word type) */
        long    ldata[N] ;/*             (long type) */
    } u ;           /* N is the number of read data */
} IODBPMC ;
```

FIG 6: iodbpmc data structure

The PMC is able to store and output different data formats at each address. If you enter cdata (lowercase without brackets!) for Output item you get byte data from the selected address, for idata you get integer and for ldata long.

# AVAILABLE PARAMETERS

Here you can find all available parameters currently implemented in the standard version of the driver:

## cnc_statinfo

Only ‘Output item’ e.g. aut, run, motion… is required, see second(!) typedef struct odbst description (section Series 160/180/210) for details.

## cnc_rddynamic2 and cnc_rddynamic

'Input parameter 1' = axis (-1 for all, 0 for none, 1-n for controlled axes)

'Output item' = item e.g. alarm or prgnum that should be output, see typedef struct odbdy2 for both functions. For axis positions you can either input faxis for all or oaxis for a specified axis.

## cnc_rdparam

To check if specific parameters are available on your controller use our tool to read parameters.

'Input parameter 1' = parameter number

'Input parameter 2' = axis (-1 for all, 0 for none, 1-n for controlled axes)

'Output item' = cdata (byte) idata (integer) ldata (long) or rdata (real data with decimal point). See cnc_rdparam for details.

## cnc_rdmacro

Only 'Input parameter 1' = macro number is needed

## cnc_modal

'Input parameter 1' = type, see cnc_modal for details. Please select types with note 'one by one'.

'Input parameter 2' = block, e.g. 0 = active block, 1 = next block, 2 = block after next block

'Output item' = Either aux, raux1, raux2

## cnc_alarm

Only 'Output item' is required. number outputs alarm number text outputs alarm text for your specific controller type. See cnc_alarm for details.

## pmc_rdpmcrng

To check if you can get data from a specific area on your controller's PMC use our tool to read PMC data.

'Input parameter 1' = area, e.g. user data (9) see pmc_rdpmcrng for details.

'Input parameter 2' = address number to be queried, e.g. 118

'Output item' = kind of data to be read, cdata = byte, idata = integer, ldata = long

### cnc_oeestate

Only ‘Output item’ is required, can be one of the following:

- statenumber = OEE state number e.g. 5 (Error)
- statetext = OEE state text e.g. Error (5)
- modenumber = Current mode number of the device e.g. 1 (Memory)
- modetext = Current mode text of the device e.g. Memory (1)

## NOTES

Please start with a simple configuration and not too much items at the same time. To do so, you can limit the number of items by simply setting Commands in section [GENERAL] of your INI file to a smaller value e.g. 3 and reload the INI file.

To query multiple different values of the same kind e.g. macros, parameters or pmc data you can simply copy and paste a similar section. Please make sure to renumber ALL your section identifiers [N] in the INI file in this case.

To add new items please adjust the number of commands in section [GENERAL] in the INI file (max 30), copy and paste a similar section to the end of the INI file, change the section number, save and reload it.

Initially use simple functions like cnc_oeestate, cnc_alarm or cnc_statinfo. You can then continue with more complex things like cnc_modal.

Please input all parameters strictly in lowercase. If you misspell an output item you get NO output. If you use an invalid command your driver informs you via the log file of the device in the \log subfolder (e.g. log_devicenumber_date.txt). Check this file when testing!

Parameters may have different output formats and may be axis related or not. For example ‘cutting feed rate’ 1410 must NOT be queried with a controlled axis but HAS to have rdata as ‘Output item’. On the other hand parameters like 1420 - 1430 REQIRE input of a controlled axis.

# TROUBLESHOOTING

If the machine shows a red icon, state disconnected and you get just output E = 0 the machine is not reachable at all. This has nothing to do with Fanuc FOCAS, it's a network issue, DNS Hostname or IP Address is wrong.

If the machine shows a green icon, state connected but you just get output E = 1 the machine is reachable but Fanuc FOCAS does not respond to commands, check your FOCAS setup and the parameters used in CNCnetPDM.ini (esp. Port number). You can also use the tool fanucdata to check if the machine replies to Fanuc FOCAS commands.

If you do not get any of the additional items please make sure that CollectOrders is set to 1 in section [General] of CNCnetPDM.ini. If just the axes position is missing set CollectFeeder to 1 in the same section. If you forced items to section 2 also set CollectQuality to 1.

If specific items are not acquired check first with fanucdata that you get these values. Also make sure that the numeric section of the item in the device drivers INI file is activated and has an entry for 'Name', inactive or items with empty names or incorrect parameters are NOT acquired.

If the machine outputs correct states but you do not see any Z (counter) values CNCnetPDM does not receive data by using the specified parameter or PMC address value, review the settings used by the machine. Also make sure that you have enabled CollectCounters = 1 in CNCnetPDM.ini