DOCUMENTATION ISG-kernel

CNC-OS-Interface API Documentation

Industrielle Steuerungstechnik GmbH
STEP, Gropiusplatz 10
70563 Stuttgart

www.isg-stuttgart.de
support@isg-stuttgart.de

RELEASE: 08/2024

Table of contents

1 Definitions

1.1 Abbreviations

Common abbreviations:

Name	Description
HLI	**H**igh **L**evel **I**nterface
MAI	**M**ultiple **A**xis **I**nterpolator
SAI	**S**ingle **A**xis **I**nterpolator

Names of CNC threads:

Name	Description	Thread
CNC_DEC	Decoding and path preparation (DEC, WRK, BAVO)	task_rnd
CNC_HMI	Communication to HMI (COM)	task_com
CNC_IPO	Velocity profile generation and interpolation (GEO)	task_int
CNC_SYS	System control functions (MMI driver)	task_mmi_driver
CNC_TCP	TCP/IP communication with HMI	task_tcp

2 CNC API functions

File name	Description
cnc_os_ifc.h	Header file with declaration of CNC API functions.

2.1 Start-up and initialization functions

2.1.1 Memory information

PLC interface:

Function name	Description	
cnc_get_sizeof_hli_platform()	Get size of HLI platform structure	
Parameter	Data type	Description
-	void	-

Function name	Description	
cnc_get_sizeof_hli_channel()	Get size of HLI channel structure	
Parameter	Data type	Description
-	void	-

Function name	Description	
cnc_get_sizeof_hli_axis()	Get size of HLI axis structure	
Parameter	Data type	Description
-	void	-

2.1.2 Register functions

2.1.2.1 Scheduling

User threads:

Function name	Description	
`cnc_register_function()`	Register functions so that they can be called by CNC threads of the CNC scheduler	
Parameter	Data type	Description
p_function	void*	Function
name	char*	Name of function

Synchronisation semaphore:

Function name	Description	
`cnc_register_sync_trigger()`	Register semaphore for CNC start tick	
Parameter	Data type	Description
hTriggerCnc	ISG_HSEM	Handle of CNC start semaphore

2.1.2.2 PLC interface

Function name	Description	
`cnc_register_hli_platform()`	Register address of HLI platform memory	
Parameter	Data type	Description
p_hli_platform	void*	Pointer to HLI platform memory

Function name	Description	
`cnc_register_hli_channel()`	Register address of HLI channel memory	
Parameter	Data type	Description
index	unsigned int	Index of channel
p_hli_channel	void*	Pointer to HLI channel memory

Function name	Description	
`cnc_register_hli_axis()`	Register address of HLI axes memory	
Parameter	Data type	Description
index	unsigned int	Index of axis
p_hli_axis	void*	Pointer to HLI axis memory

Function name	Description	
`cnc_register_external_variables_platform()`	Register address of external variables platform memory	
Parameter	Data type	Description
p_memory	void*	Pointer to global VE memory
length	unsigned int	Length of memory

Function name	Description	
`cnc_register_external_variables_channel()`	Register address of external variables channel memory	
Parameter	Data type	Description
index	unsigned int	Index of channel
p_memory	void*	Pointer to global VE memory
length	unsigned int	Length of memory

2.1.2.3 Drive interface

Function name	Description	
`cnc_register_drive_command_interface()`	Register address of drive command interface memory	
Parameter	Data type	Description
index	unsigned int	Index of drive
drive_command	void*	Pointer to input memory
length	unsigned int	Length of memory

Function name	Description	
`cnc_register_drive_feedback_interface()`	Register address of drive feedback interface memory	
Parameter	Data type	Description
index	unsigned int	Index of drive
drive_feedback	void*	Pointer to output memory
length	unsigned int	Length of memory

Function name	Description	
`cnc_register_serc_write_req_function()`	Register callbacks for sercos service channel write requests.	
Parameter	Data type	Description
p_function	int	Callback function
Callback Parameter	Data type	Description
invokeID	unsigned long	Unique invoke ID of the request. Must be provided during the response call so that it can be assigned to the request.
logAxisNo	long	Logical axis number of the corresponding drive.

sercID	unsigned long	Sercos element:
		`DATASTATE = 0x0000`
		`NAME = 0x0010`
		`ATTRIBUTE = 0x0018`
		`UNIT = 0x0020`
		`MIN = 0x0028`
		`MAX = 0x0030`
		`VALUE = 0x0038`

size	unsigned long	Size of the data.
pData	void*	Pointer to the data.

Function name	Description	
`cnc_register_serc_read_req_function()`	Register callbacks for sercos service channel read requests.	
Parameter	Data type	Description
p_function	int	Callback function
Callback Parameter	Data type	Description
invokeID	unsigned long	Unique invoke ID of the request. Must be provided during the response call so that it can be assigned to the request.
logAxisNo	long	Logical axis number of the corresponding drive.

sercID	unsigned long	Sercos element:
		`DATASTATE = 0x0000`
		`NAME = 0x0010`
		`ATTRIBUTE = 0x0018`
		`UNIT = 0x0020`
		`MIN = 0x0028`
		`MAX = 0x0030`
		`VALUE = 0x0038`

size	unsigned long	Size of the data.

Function name	Description	
`cnc_serc_service_channel_write_res()`	Interface method to respond to a service channel write request.	
Parameter	Data type	Description
invokeID	unsigned long	Unique invoke ID of the request. Must be provided during the response call so that it can be assigned to the request.
result	unsigned long	Result of the write request.

Function name	Description	
`cnc_serc_service_channel_read_res()`	Interface method to respond to a service channel read request.	
Parameter	Data type	Description
invokeID	unsigned long	Unique invoke ID of the request. Must be provided during the response call so that it can be assigned to the request.
result	unsigned long	Result of the read request.

size	unsigned long	Size of the data.
pData	void*	Pointer to the data.

2.1.3 CNC start-up

Function name	**Description**	
`cnc_pre_initialize()`	Call user function that must be done before start-up of CNC, e.g. license check.	
Parameter	Data type	Description
-	void	-

Function name	**Description**	
`cnc_startup()`	Start-up function of CNC: - start CNC threads - allocate CNC internal memory - initialize all CNC data	
Parameter	Data type	Description
cnc_startup_file	char*	Path and file name of CNC start-up file

Function name	**Description**	
`cnc_set_xml_parser_path()`	At the start of the CNC start-up, an attempt is made to load the libxml2 library dynamically. This is usually already available on Linux systems. On Windows systems, this method can be used to specify the path to the corresponding dll before the start-up.	
Parameter	Data type	Description
libxml2_path	char*	Path and file name of the dynamic libxml2 library

2.2 Scheduling

The CNC Kernel has the capability to use various types of scheduling. Firstly, there is an internal scheduler which can be configured in various ways. Additionally, through external scheduling, the CNC Kernel can easily be integrated into any control system. Moreover, both types of scheduling can be operated in combination.

For the operation of the CNC Kernel, the following three tasks are required:

- **CNC_IPO**
  Interpolator Task
- **CNC_DEC**
  Decoder Task
- **CNC_HMI**
  Communication Task for HMI Connection

Additionally, there are the following tasks:

- **CNC_SYS**
  SYS Driver for AHMI Communication
- **CNC_TCP**
  TCP/IP Communication

2.2.1 Types of Scheduling

2.2.1.1 Internal Scheduling

Internal scheduling is configured via the rtconf.lis file. For configuration, refer to the relevant documentation Configuration of Real-Time Parameters.

For the operation of the internal scheduler, either a timer can be configured, or it can be triggered via an external trigger signal (semaphore).

The semaphore can be created using the API function `cnc_create_semaphore()` and triggered using the function `cnc_give_semaphore()`.

2.2.1.2 Externes Scheduling

The rtconf.lis configuration file is not required for external scheduling.

Before calling the `cnc_startup()` function, the cycle time must be set in the CNC kernel. The function `cnc_set_cycle_time()` is used for this purpose, and the parameter to be passed is specified in microseconds (µs).

```
cnc_set_cycle_time(1000);
```

To access the task functions, the function `cnc_lookup_function()` is required. It provides function pointers to the respective task functions, which can then be used for external scheduling. `cnc_lookup_function()` provides valid function pointers only after `cnc_startup()` has been successfully executed.

```
typedef int (*task_IPO_t)  (int(*p_function)(unsigned long));
typedef int (*task_DEC_t)  (int(*p_function)(unsigned long));
typedef int (*task_HMI_t)  (int(*p_function)(unsigned long));
typedef int (*task_SYS_t)  (int(*p_function)(unsigned long));
typedef int (*task_TCP_t)  (int(*p_function)(unsigned long));

task_IPO_t  task_IPO;
task_DEC_t  task_DEC;
task_HMI_t  task_HMI;
task_SYS_t  task_SYS;
task_TCP_t  task_TCP;

task_IPO = (task_IPO_t)cnc_lookup_function((char*)"task_int");
task_DEC = (task_DEC_t)cnc_lookup_function((char*)"task_rnd");
task_HMI = (task_HMI_t)cnc_lookup_function((char*)"task_com");
task_SYS = (task_SYS_t)cnc_lookup_function((char*)"task_mmi_driver");
task_TCP = (task_TCP_t)cnc_lookup_function((char*)"task_tcp");
```

For the operation of the CNC Kernel, the mentioned functions are then cyclically called.

```
while (f_cnc_shutdown == false)
{
  ...
  task_DEC(0);
  task_DEC(0);
  task_HMI(0);
  task_IPO(0);
  cnc_drive_simulation_ipo(0);
  task_HMI(0);
  task_SYS(0);
  task_TCP(0);
  ...
}
```

2.2.1.3 Mixed Mode

It is also possible to use a combination of internal and external scheduling.

For this, the tasks assigned to the internal scheduler are configured accordingly in `rtconf.lis`. Otherwise, proceed as described above.

It should be noted that the parameter `P-RTCF-00001 (interrupt_source)` in `rtconf.lis` contains the following value:

```
interrupt_source              3     # external Semaphore
```

Also, the cycle time for scheduling should be entered in `rtconf.lis`. Setting this via the API function `cnc_set_cycle_time()` is omitted at this point.

2.2.2 API functions

Function name	**Description**	
`cnc_schedule_threads()`	Direct call of CNC scheduler (if semaphore for CNC start tick is not used).	
Parameter	Data type	Description
-	void	-

Function name	**Description**	
`cnc_create_semaphore()`	Initialize and open a semaphore.	
Parameter	Data type	Description
hsem	ISG_HSEM	Pointer to Semaphore
name	char*	Name of the semaphore

Function name	**Description**	
`cnc_give_semaphore()`	Give (unlock) semaphore.	
Parameter	Data type	Description
hsem	ISG_HSEM	Semaphore

Function name	**Description**	
`cnc_delete_semaphore()`	Close semaphore.	
Parameter	Data type	Description
hsem	ISG_HSEM	Semaphore

Function name	**Description**	
`cnc_lookup_function()`	Get function pointer of registered function. Function pointer has to be of type `typedef int (*p_func_t) (int(*p_function)(unsigned long));`	
Parameter	Data type	Description
name	char*	Name of the registered function

Function name	Description	
`cnc_set_cycle_time()`	Used to set the NC cycle time in microseconds. This is usually defined by the configuration list of the real-time parameters (P-RTCF-00002). If all CNC tasks are called directly during external scheduling, this value can be set using this function. The configuration list of the real-time parameters can be omitted in this case.	
Parameter	Data type	Description
time_in_us	unsigned int	Cnc cycle time

2.3 Shutdown functions

Function name	Description	
`cnc_shutdown()`	Shutdown of CNC: - stop all CNC threads - deallocate all memory	
Parameter	Data type	Description
-	void	-

2.4 CNC objects

2.4.1 Structure

Hierarchic structure of the CNC objects. **Platform**, **Axes**, **Channels** and **SAI** are the object groups for which the CNC objects are accessible.

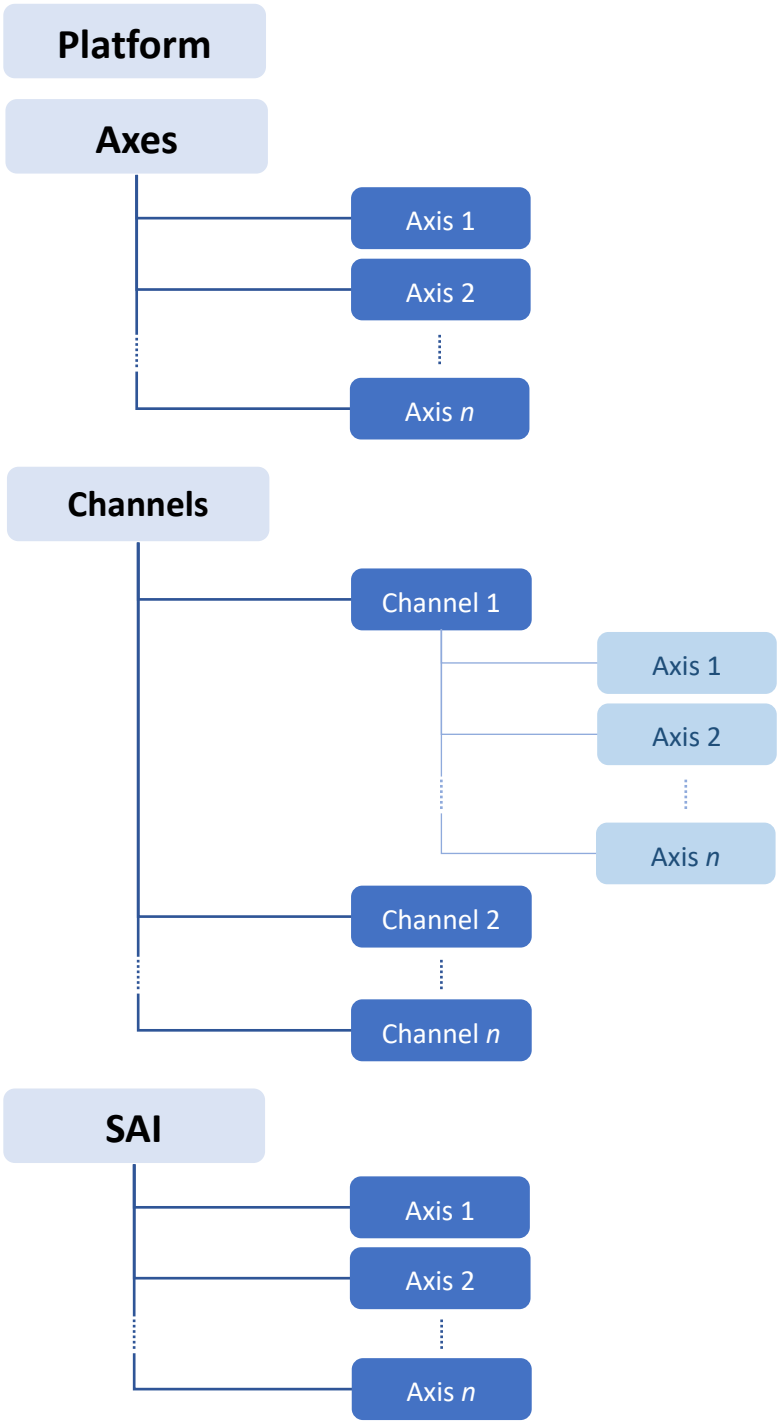

Figure 1: Hierarchic structure of the CNC objects

2.4.2 Data structures and enumerations

2.4.2.1 Data structure *CNC_OBJECT*

Defines the structure of a CNC object:

```
typedef struct _cnc_object
{
  CNC_OBJECT_ID         ObjectID;
  char                  Name[256];
  E_CNC_OBJECT_TYPE     Type;
  unsigned long         Length;
  char                  Unit[128];
  bool                  Writeable;

} CNC_OBJECT;
```

2.4.2.2 Data structure *CNC_OBJECT_ID*

Defines the data structure of a CNC object id:

```
typedef struct _cnc_object_id
{
  unsigned long         iThread;
  unsigned long         iGroup;
  unsigned long         iOffset;

} CNC_OBJECT_ID;
```

2.4.2.3 Enumeration *E_CNC_OBJECT_TYPE*

Defines the data types for the CNC objects:

- CNC_OBJECT_TYPE_NONE
- CNC_OBJECT_TYPE_BOOLEAN
- CNC_OBJECT_TYPE_UNS08
- CNC_OBJECT_TYPE_SGN08
- CNC_OBJECT_TYPE_UNS16
- CNC_OBJECT_TYPE_SGN16
- CNC_OBJECT_TYPE_UNS32
- CNC_OBJECT_TYPE_SGN32
- CNC_OBJECT_TYPE_UNS64
- CNC_OBJECT_TYPE_SGN64
- CNC_OBJECT_TYPE_REAL64
- CNC_OBJECT_TYPE_STRUCT
- CNC_OBJECT_TYPE_REAL32
- CNC_OBJECT_TYPE_CHAR
- CNC_OBJECT_TYPE_STRING
- CNC_OBJECT_TYPE_ERROR

2.4.2.4 Enumeration *E_CNC_OBJECT_DOMAIN*

Defines the different objects groups:

- `CNC_PLATFORM`
- `CNC_CHANNEL`
- `CNC_AXIS`
- `CNC_SAI`

2.4.2.5 Enumeration *E_CNC_THREAD_ID*

Defines the threads the CNC objects are assigned to.

- `CNC_TASK_IPO = 1`
- `CNC_TASK_DEC = 2`
- `CNC_TASK_HMI = 3`

2.4.3 Access functions

Function name	**Description**	
`cnc_set_tcp_object_access ()`	This function can be used to set whether the CNC objects are accessed via TCP. If the CNC objects are not accessed via TCP, only synchronous access to the CNC object from the corresponding task context is permitted. If access is made via TCP, access does not necessarily have to be made from the corresponding task context.	
Parameter	Data type	Description
tcp_access	bool	True: TCP access is activated False: TCP access is deactivated

2.4.3.1 Common access functions

Please note: each CNC object is assigned to a thread (iThread) and access to it (read/write) has to be done in the context of this thread. For this, use some user defined function which is assigned to one of the main thread(IPO, DEC, HMI).

Function name	**Description**
`cnc_query_number_of_objects()`	Returns the number of available kernel objects
Returned data type	Description
int	Number of available CNC objects in the given context

Function name	**Description**
`cnc_query_object()`	Returns the object description of the given index
Returned data type	Description

int	Error return value	
Parameter	Data type	Description
index	unsigned int	Object index
cnc_object	CNC_OBJECT *	Pointer on object description data structure

Function name	Description	
`cnc_query_object_domain()`	Returns the object domain of the given object id	
Returned data type	Description	
E_CNC_OBJECT_DOMAIN	Object domain of the given object index. Returns CNC_NONE if no object domain is found.	
Parameter	Data type	Description
cnc_object	CNC_OBJECT_ID	CNC object id for which the object domain is to be returned

Function name	Description	
`cnc_read_value()`	Reads the value of the given object id within the context in which the function is called	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	void *	Pointer to the object value
length	unsigned int	Length of the read value

Function name	Description	
`cnc_write_value()`	Write a value to the given object id within the context in which the function is called	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	void *	Pointer to the object value to be written
length	unsigned int	Length of the value to be written

Function name	Description	
`cnc_read_write_value()`	Read and write functionality for an object. Is generally required to transfer parameters for the read process - e.g. variable names when reading a decoder variable.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id

value	void *	Pointer to the object value
Read length	unsigned int	Admissable length of the read value
Write length	unsigned int	Length of the write value

2.4.3.2 Data type specific acces functions

Function name	**Description**	
`cnc_read_BOOL()`	Reads the boolean value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	bool *	Pointer on the object value to be returned

Function name	**Description**	
`cnc_write_BOOL()`	Writes the boolean value of the given object	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	bool	Value to be written

Function name	**Description**	
`cnc_read_CHAR()`	Reads the char value of the given object id.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	char *	Pointer on the object value to be returned

Function name	**Description**	
`cnc_write_CHAR()`	Writes the char value of the given object	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description

object_id	CNC_OBJECT_ID	CNC object id
value	char	Value to be written

Function name	Description	
`cnc_read_SGN08()`	Reads the signed char value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed char *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_SGN08()`	Writes the signed char value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed char	Value to be written

Function name	Description	
`cnc_read_UNS08()`	Reads the unsigned char value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned char *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_UNS08()`	Writes the unsigned char value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned char	Value to be written

Function name	Description	
`cnc_read_SGN16()`	Reads the signed short value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed short *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_SGN16()`	Writes the signed short value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed short	Value to be written

Function name	Description	
`cnc_read_UNS16()`	Reads the unsigned short value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned short *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_UNS16()`	Writes the unsigned short value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned short	Value to be written

Function name	Description	
`cnc_read_SGN32()`	Returns the signed int value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed int *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_SGN32()`	Writes the signed int value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	signed long	Value to be written

Function name	Description	
`cnc_read_UNS32()`	Reads the unsigned int value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned int *	Pointer on the object value to be returned

Function name	Description	
`cnc_write_UNS32()`	Writes the unsigned int value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	unsigned long	Value to be written

Function name	Description	
cnc_read_REAL64()	Reads the double value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	double *	Pointer on the object value to be returned

Function name	Description	
cnc_write_REAL64()	Writes the double value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	double	Value to be written

Function name	Description	
cnc_read_SGN64()	Reads the int 64 bit value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	int64_t *	Pointer on the object value to be returned

Function name	Description	
cnc_write_SGN64()	Writes the int 64 bit value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	int64_t	Value to be written

Function name	Description	
`cnc_read_UNS64()`	Reads the unsigned int 64 bit value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	uint64_t *	Pointer on the value to be returned

Function name	Description	
`cnc_write_UNS64()`	Writes the unsigned int 64 bit value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	uint64_t	Value to be written

Function name	Description	
`cnc_read_STRING()`	Reads the string value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	char *	Pointer on the value to be returned
length	unsigned int	Length of the read value

Function name	Description	
`cnc_write_STRING()`	Writes the string value of the given object id	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
object_id	CNC_OBJECT_ID	CNC object id
value	char *	Pointer to the value to be written
length	unsigned int	Length of the value to be written

2.5 Message logging

2.5.1 Enumeration E_CNC_LOG_MSG_CLASS

Defines the different classes of log messages:

```
typedef enum _e_log_msg_class
{
  LOG_INFO = 0,  // Message out of NC-program (#MSG)
  LOG_WARNING,   // CNC warning
  LOG_ERROR,     // CNC error message
  LOG_EXCEPTION, // system error, exception
  LOG_DEBUG      // debug prints for diagnosis by experts


} E_CNC_LOG_MSG_CLASS;
```

2.5.2 Register function

Function name	Description	
cnc_register_log_message_function()	Function to register a log message output function that is called by ISG-kernel to transport error messages, warnings, messages (#MSG) and debug prints.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
p_function	int(*)(E_CNC_LOG_MSG_CLASS, unsigned int, const char*)	Log message output function

Function name	Description	
cnc_register_log_message_json_function()	Function to register a log message output function in that is called by ISG-kernel to transport error information. The information is provided via json stream.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
p_function	int(*)(const char*)	Log message output function

2.5.3 Setting function

Function name	Description

cnc_set_startup_error_log()	Activates logging during CNC startup.	
Parameter	Data type	Description
startup_error_log	bool	True: File logging during startup activated False: Logging deactivated

Function name	**Description**	
cnc_startup_error_log_file ()	Sets the path and file name of the CNC startup logging file.	
Parameter	Data type	Description
log_file_name	char*	Path and file name of CNC start-up logging file

2.5.4 Runtime function

Function name	**Description**	
cnc_log_message()	Log message output function that is called by ISG-kernel to transport error messages, warnings, messages (#MSG) and debug prints.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
LogMsgClass	E_CNC_LOG_MSG_CLASS	Log message class
LogMsgId	unsigned int	ID of the log message
LogMsgString	char*	String which contains the log message

2.6 External Tool Management (tool change)

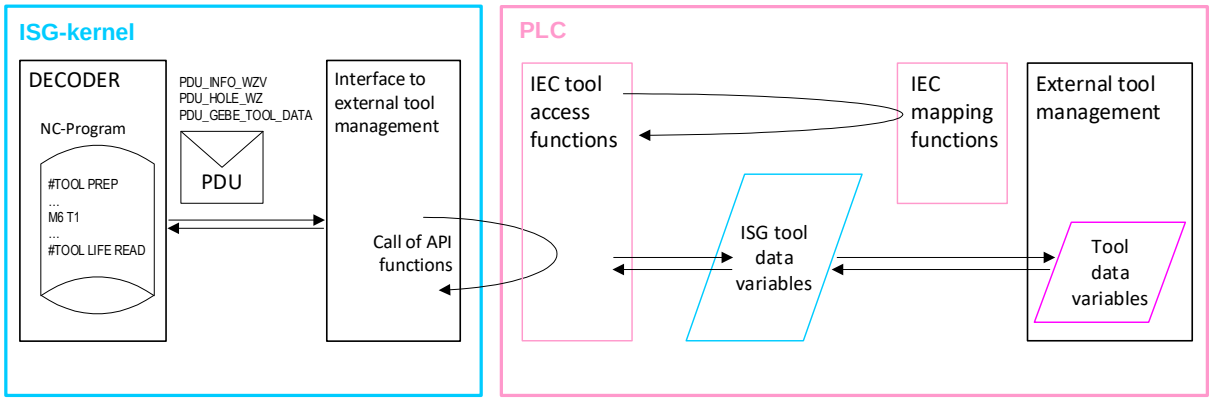

Figure 2: Tool data exchange of ISG kernel to external tool management in PLC

File name	Description
`cnc_tool_ifc.h`	Header file with ISG-kernel tool data description data structure (CNC_TOOL_DESC)

2.6.1 Memory information

Function name	**Description**	
`cnc_get_sizeof_cnc_tool_desc()`	Get size of CNC tool description structure	
Parameter	Data type	Description
-	void	-

2.6.2 Register functions

Function name	**Description**	
`cnc_register_tool_change_info()`	Register function cnc_tool_change_info() in ISG-kernel	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
p_function	int *	Function

Function name		**Description**	
`cnc_register_read_tool_data()`		Register function cnc_read_tool_data() in ISG-kernel	
Returned data type	Description		
int	Error return value		
Parameter		Data type	Description
p_function		int *	Function

Function name		**Description**	
`cnc_register_write_tool_data()`		Register function cnc_write_tool_data() in ISG-kernel	
Returned data type	Description		
int	Error return value		
Parameter		Data type	Description
p_function		int *	Function

2.7 Return codes of CNC API functions

Define	Value	Description
ERR_CNC_DELETE_SHM_HLI_AXIS	-55	Error while deleting axis shared memory.
ERR_CNC_DELETE_SHM_HLI_CHANNEL	-54	Error while deleting channel shared memory.
ERR_CNC_DELETE_SHM_HLI_PLATFORM	-53	Error while delete platform shared memory.
ERR_CNC_CREATE_SHM_HLI_AXIS	-52	Error while creating/getting axis shared memory.
ERR_CNC_CREATE_SHM_HLI_CHANNEL	-51	Error while creating/getting channel shared memory.
ERR_CNC_CREATE_SHM_HLI_PLATFORM	-50	Error while creating/getting platform shared memory.
ERR_CNC_PATH_TOO_LONG	-41	Path is too long.
ERR_CNC_READ_TCPIP	-40	Remote call via TCP/IP not possible
ERR_CNC_INVALID_AX_NUMBER	-35	Invalid axis number
ERR_CNC_INVALID_CH_NUMBER	-34	Invalid channel number
ERR_CNC_STARTUP_TIMEOUT	-33	Timeout during CNC startup
ERR_CNC_STARTUP_FILE_NOT_FOUND	-32	CNC startup file not found
ERR_CNC_TOOL_DATA_PTR_INVALID	-31	Pointer to external tool data is invalid
ERR_CNC_LICENSE_CHECK_FAILED	-30	License check failed
ERR_CNC_HLI_SHM_PTR_INVALID	-29	Pointer to HLI is invalid
ERR_CNC_HLI_SHM_ALLOCATE	-28	Allocation of shared memory for HLI failed
ERR_CNC_DRIVE_FEEDBACK_PTR_INVALID	-27	Pointer to drive feedback memory is invalid
ERR_CNC_DRIVE_CMD_PTR_INVALID	-26	Pointer to drive command memory is invalid
ERR_CNC_EXT_VAR_PTR_INVALID	-25	Pointer to external variables memory is invalid
ERR_CNC_HLI_AXIS_PTR_INVALID	-24	Pointer to HLI axis memory is invalid
ERR_CNC_HLI_CHANNEL_PTR_INVALID	-23	Pointer to HLI channel memory is invalid
ERR_CNC_HLI_PLATFORM_PTR_INVALID	-22	Pointer to HLI platform memory is invalid
ERR_CNC_ERROR_INITIALIZING_HLI	-21	Error while initializing HLI
ERR_CNC_ERROR_UPDATING_PARAMETERS	-20	Parameter update failed
ERR_CNC_INVALID_TOOL_NR	-19	Tool number not valid or does not exist
ERR_CNC_INVALID_AX_INDEX	-18	Axis index not valid or does not exist
ERR_CNC_INVALID_CH_INDEX	-17	Channel index not valid or does not exist
ERR_CNC_EXT_TOOL_MGR_INIT_FAILED	-16	Initialization of external tool management failed
ERR_CNC_MEMORY_REG_FAILED	-15	Registration of CNC memories failed.
ERR_CNC_PTR_UEBER_NOT_AVAILABLE	-14	Pointer to global CNC shared memory invalid

Define	Value	Description
ERR_CNC_NOT_SUFFICIENT_MEMORY	-13	Not sufficient memory
ERR_CNC_INVALID_DRIVE_IFC	-12	Drive interface not available
ERR_CNC_SCHEDULER_UNKNOWN_FCT	-11	Unknown function for CNC scheduler
ERR_CNC_TOOL_FCT_NOT_INIT	-10	External tool function not available
ERR_CNC_MSG_FCT_NOT_INIT	-9	External message function not available
ERR_CNC_WRONG_OBJECT_DATA_TYPE	-8	Object data type wrong
ERR_CNC_WRONG_OBJECT_INDEX	-7	Object index wrong
ERR_CNC_INVALID_OBJECT_ID	-6	Object ID wrong
ERR_CNC_OBJECT_NO_FOUND	-5	Object not found
ERR_CNC_READ_WRITE_SIZE_TO_BIG	-4	Size of object too big
ERR_CNC_OBJECT_DOMAIN_NOT_AVAILABLE	-3	Object domain not available
ERR_CNC_INVALID_INDEX_GROUP	-2	Object group index not valid
ERR_CNC_INVALID_TASK_ID	-1	Object task ID not valid
ERR_CNC_NOERROR	0	No error

2.8 Drive Interface Simulation

File name	Description
cnc_demo_drive_simulation.c	Simulation functions

Function name	Description	
cnc_demo_drive_ifc_simulation_coe()	Simulation of synchronous commounication for CAN over EtherCAT.	
Parameter	Data type	Description
i_drive	int	Index of drive interface
p_drive_command	CNC_DRIVE_COMMAND_COE*	Pointer to drive command interface memory
p_drive_feedback	CNC_DRIVE_STATUS_COE *	Pointer to drive feedback interface memory

Function name	Description	
cnc_demo_drive_ifc_simulation_soe()	Simulation of synchronous and asynchronous commounication for SERCOS over EtherCAT.	
Parameter	Data type	Description
i_drive	int	Index of drive interface
p_drive_command	CNC_DRIVE_COMMAND_SOE*	Pointer to drive command interface memory
p_drive_feedback	CNC_DRIVE_STATUS_SOE *	Pointer to drive feedback interface memory

2.9 PLC Connection

Communication between CNC Kernel and a PLC is established via High Level Interface (HLI). The interface is mapped to named shared memories. These shared memories may be created either in the CNC Kernel or in the PLC.

For this, the CNC API provides the following functions:

2.9.1 API Functions

Function name	**Description**	
`cnc_create_plc_interfaces()`	Created and register named shared memories for HLI connection.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
number_of_channels	unsigned int	Number of configured channels.
number_of_axes	unsigned int	Number of configured axes.

Function name	**Description**	
`cnc_connect_plc_interfaces()`	Get and register named shared memories for HLI connection.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
number_of_channels	unsigned int	Number of configured channels.
number_of_axes	unsigned int	Number of configured axes.

Function name	**Description**	
`cnc_delete_plc_interfaces()`	Delete named shared memories for HLI connection.	
Returned data type	Description	
int	Error return value	
Parameter	Data type	Description
number_of_channels	unsigned int	Number of configured channels.
number_of_axes	unsigned int	Number of configured axes.

2.9.2 Shared Memory Names

Names for shared memories are fixed and may not be changed.

Name	Beschreibung
SHM_HLI_PLATFORM.DAT	Named shared memory for HLI plaform.
SHM_HLI_CHANNEL_*X*.DAT	Named shared memory for the HLI channel. *X* stands for the channel index.
SHM_HLI_AXIS_X.DAT	Named shared memory for die HLI axes. *X* stands for the axis index.