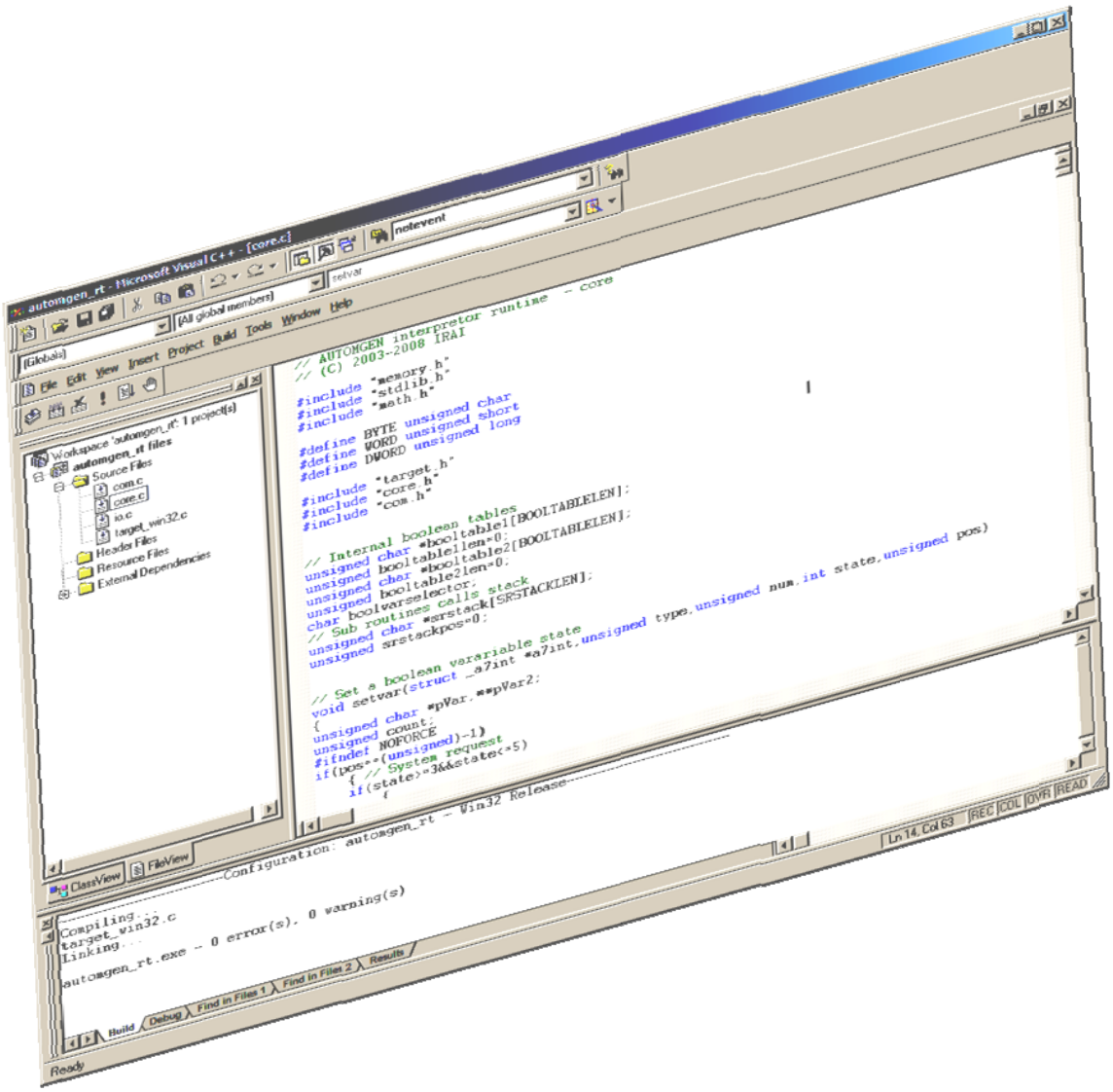
AUTOMGEN RUNTIME

C source Runtime for AUTOMGEN

Open Source



2008 IRAI - www.irai.com

The purpose is to propose a free AUTOMGEN runtime which can be used to build AUTOMGEN compatible targets. The runtime itself is a very small program which is able to run directly AUTOMGEN pivot code generated with AUTOMGEN INT post-processor. The RUNTIME is able to run 100% of the AUTOMGEN programming, SCADA and WEB SCADA functionalities. AUTOMGEN is able to communicate with the RUNTIME on serial communication port or on TCP-IP.

AUTOMGEN runtime is composed by small c source files. This is a low level concept, so it should run on targets with low resources.

Files are classified in two groups:

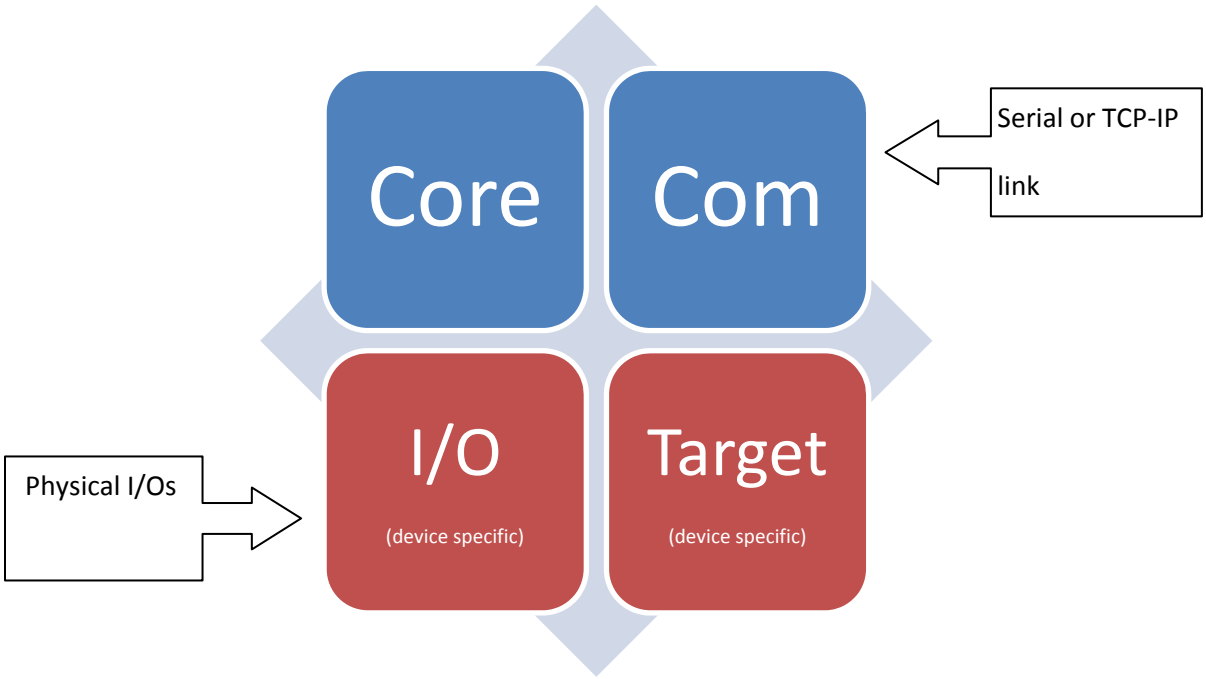
- target specific group (files must be modified for each targets)
- standard group (files should be compiled without modifications for any targets).

The included samples can be compiled with:

- VISUAL C 6.0, .NET 2003 or .NET 2005 for Windows 32 bits samples,

- embedded Visual C++ 4.0 for Windows CE,

- the standard files should be compiled with any C compilers.



Standard Files

Core.c

This is the main part of the runtime; this part is able to run AUTOMGEN programs.

Com.c

This is the high level communication part. The low level is device specific (target_xxx.c). The Com.h header file contains the communication settings: serial port (number, speed, parity, etc.) or TCP-IP (port number).

Specific Files

Io.c

This is the link between the runtime and the physical I/Os.

The functions of this module must be modified for driving the physical I/O available on the device (see samples below).

List of the io.c functions

int initio(void)

Must initialize the physical I/O. Must return 0 if the I/O initialization is ok, otherwise <0.

Sample:

```
int initio(void)
{
if(initializemyio()<0) return -1;
return 0;
}
```

int uninitio(void)

Must uninitialize the physical I/O. Must return 0 if ok, otherwise <0.

int readi(struct _a7int *a7int)

Must read physical inputs and transfer data into AUTOMGEN variables. Must return 0 if ok, otherwise <0.

Sample (assuming that geti function returns 32 digital inputs in a dword and getai function returns a 16 bits analog value in a word).

```
int readi(struct _a7int *a7int)
{
unsigned count;
unsigned long digitali;
digitali=geti();
// 32 digital inputs
for(count=0;count<32;count++)
        {
        setvar(a7int,0,a7int->io_i_pos++,(digitali&(1<<count))?1:0,0xffffffff);
```

```
        }
// 1 analog input
a7int->pM[a7int->io_m_pos++]=getai();
return 0;
}
```

Notes:

1- setvar is a core function which must be used for Boolean variables state modifications. The direct access is not permitted for writing Boolean variable values.
2- io_i_pos, io_o_pos and io_m_pos members can be used for accessing the next variables.

int writeo(struct _a7int *a7int)

Must write physical outputs from the states of the AUTOMGEN variables. Must return 0 if OK, otherwise <0.

Sample (assuming puto writes 16 digital outputs of 4 different outputs modules and putao writes 4 analog values).

```
int writeo(struct _a7int *a7int)
{
unsigned modulecount;
unsigned count;
// 4 x 16 digital outputs
for(modulecount=0;modulecount<4;modulecount++)
        {
        unsigned short v=0;
        for(count=0;count<16;count++)
                {
                if(a7int->pO[a7int->io_o_pos++]) v|=1<<count;
                }
        puto(modulecount,v);
        }
// 4 analog values (assuming putao need a pointer to 4 16 bits values)
putao(&pM[a7int->io_m_pos]);
a7int->io_m_pos+=4;
return 0;
}
```

Target_xxx.c

This is a target specific file, the provided files target_xxx.c file show some specifics files:

Target_win32_console : for windows 32 OS in console mode

Target_win32_gui : for windows 32 OS gui mode

Target_wince : for Windows CE OS

List of the target_xxx.c functions

DWORD target_gettimeinms(void)

Must return a DWORD which is a 32 bits time counter in ms. The starting value does not matter, the important is that the returned value between two calls evolves regarding the number of 1/1000 seconds which have elapsed.

void target_readrtc(unsigned short *ms,unsigned short *sec,unsigned short *min,unsigned short *hour,unsigned short *day,unsigned short *month,unsigned short *year)

Must return the current date and time in the 7 unsigned shorts.

void target_fatal(void)

Called when a fatal error occurs, by example the execution of an invalid operation code.

char *target_id(void)

Must return a 16 chars string (the length must be exactly 16 chars) which is the name of the target. This name will be displayed in AUTOMGEN at connection time.

unsigned char target_confbyte(void)

Must return a configuration byte. Actually only bit 1 (value = 2) is used. This bit must be set to 0 if the integers are coded with LSB first, otherwise set this bit to 1.

int target_com_open(unsigned port,unsigned speed,unsigned parity,unsigned databits,unsigned stopbits)

Open a serial port for communications.
Port is the number of the port, 1=first port, 2=second, etc.
Speed is the baud rate in bauds,
Parity is one of these chars : N (for none), O (for odd), E (for even),
Databits: always 8.
Stopbits: 1 or 2.

This function must return 0 for success, <0 for error.

void target_com_send(unsigned char *buff,unsigned len)

This function must send the len chars of the buff buffers to the serial link.

int target_com_getchar(void)

This function must return an incoming char from the serial link or -1 if no char has been received.

void target_com_close(void)
This function must close the serial link.

int target_netcom_start(void)
This function must initialize the TCP-IP connection and must return 0 for success or -1 otherwise.

int target_netcom_close(void)
This function must close the TCP-IP connection.

int target_netcom_srvread(unsigned client,unsigned char *data,unsigned maxdata)
This function must return datas read from the specified TCP-IP client. Must return the number of read bytes in case of success, otherwise -1.

int target_netcom_srvgetlenin(unsigned client)
Must return the number of available chars for the specified TCP-IP client or -1 if error.

int target_netcom_srvgetfirstclient(void)
Must return the first TCP-IP client connected or -1 if no client is connected.

int target_netcom_srvgetnextclient(unsigned cli)
Must return the next TCP-IP client (following cli) connected or -1 if no more clients.

int target_netcom_srvopen(int (*netevent)(struct _NE *pne),WORD port)
Must create a TCP-IP server.
Netvent: always null.
Port number: port used for listening incoming client requests.

Must return 0 for success, otherwise <0.

int target_netcom_srvclose(void)
Must stop the TCP-IP server. Must return 0 for success, otherwise <0.

int target_netcom_srvsenddata(int client,BYTE *data,int len)
Must send len bytes to the specified client. Must return 0 for success, otherwise <0.

Assistance

Stephane MASSART

sm@irai.com

Tel +33 4 66 54 91 30