Applying a Real-Time Analysis Methodology to the Software of a Robot Controller

Aplicación de una metodología de análisis de tiempo real
al software del controlador de un robot

Keywords:
Real-Time Systems, Scheduability Analysis, Worst-Case Execution Time Estimation

Jakob Lombacher

May 17, 2009

Resumen

En los sistemas de tiempo real la bondad de los resultados depende del cumplimiento de ciertos parámetros temporales. Existen técnicas de análisis que permiten obtener los tiempos de respuesta de peor caso de las actividades del sistema una vez que se conocen los tiempos de ejecución del código que las realiza. Las medidas de los tiempos de ejecución en el peor caso son muy difíciles de obtener incluso midiendo, pero últimamente van apareciendo técnicas que combinan la medida de tiempos con el análisis estructural del código y que mejoran la bondad d e las medidas. Éste es el caso de la herramienta RapiTime que actualmente está operativa para medir tiempos en código fuente para los lenguajes C y Ada sobre el sistema operativo MaRTE OS.

El objetivo de este trabajo es demostrar el uso de una metodología de cálculo de los tiempos de ejecución y respuesta de peor caso en una aplicación real correspondiente al software del controlador de un robot industrial. Para ello, se hace un análisis del código fuente (en lenguaje Ada) del robot bajo estudio con objeto de identificar los segmentos de programa de los que se necesita conocer su tiempo de ejecución de peor caso. Para la obtención de los tiempos de ejecución se emplea la herramienta RapiTime sobre el sistema operativo MaRTE OS en el que se basa el software del robot. Después se hace el modelado del controlador de acuerdo al modelo MAST, que permite la aplicación de herramientas de cálculo de tiempos de respuesta para verificar si se cumplen los requisitos de tiempo real. También se comparan los resultados obtenidos con los que se habían obtenido previamente en la evaluación de los tiempos de ejecución haciendo uso de simples lecturas del reloj. Esto permite evaluar la metodología completa y sacar conclusiones.

Contents

Chapter 1

Introduction

The usual size of industrial control applications has increased in the last few years very much, both in number of lines of code and complexity of the hardware structure. Therefore, a useful approaches is needed for checking real-time abilities. This practical report deals with the evaluation of the real-time behavior of the six axes Robot CASEVA. For this purpose two tools are used, Rapitime form Rapita Systems Ltd.[7], which is a tool for measuring the (Overall Worst Case Execution Time) O-WCET in a statistical manner, and MAST [1] to observe the upper response time. MAST is an Open Source tool which was developed by the Computadores y Tiempo Real Group of the University of Cantabria.

This work starts with a brief Introduction to the real time problematic and a categorization of the measurement methods. The chapter 2 gives an overview of the given platform and the tools used, chapter 3 comprising the whole measurement including the calculations with MAST. Finally, the chapter 4 evaluates the whole process and gives some suggestions for future improvement.

1.1 Real-Time Issues

If we speak of Real-Time systems, we speak about systems that guarantee us a correct response within a given time bound. We can distinguish between two kinds of Real-Time System, the hard real-time and the soft real-time systems. Soft real-time only guarantees the behavior for most of the cases, and is used for example for multimedia applications, where not fulfilling the condition has no further consequences.

In control applications we usually have hard real-time requirements, where the time bound must be satisfied in every case. A usual control software has many branches, loops, exceptions etc. The response time is therefore dependent on the worst-case execution path. The challenge for real-time improvement is to identify this worst case.

1.2 Measurement of Execution Times

The time analysis techniques can be categorized in static methods and measurement-based methods.

Static analysis examines the control-flow graph to figure out the worst-case path(s), which usually depends on the input data and the initial state. Big challenges with this approach are dynamic jumps and the behavior of modern processors with caches and pipelines.

The other strategy is measurement based. Here, the execution time will be measured while executing the code on a processor or on a clock-cycle accurate simulator. The capture of the time traces can either happen with special debugging hardware or with some additions to the source code of the application. Therefore, it is important to attain the execution with the worst-case initial state and the worst-case input. Due to the fact that these data are usually unknown, only a test of all possibilities can really check hard real-time circumstances.

The static method emphasises safety, but often with insufficiently documented hardware the analyses are difficult, if not impossible. The measurement-based method has the problem that the worst case has to occur in the measurements. Therefore, large test scales are needed, which makes it useless for large systems.

The approach of Rapitime therefore offers an interesting solution, which combines structural analyses with measurement data. A Further description follows in 2.2.

1.3 Scheduleability Analysis

The schedulability analysis of a system tries to determine the schedulability of the system, which means that the system meets the timing requirements specified for it (normally in terms of deadlines). A lot of off-line and on-line algorithms exist to ensure the schedulability of a mono-processor or even a distributed system [8]. Some of the algorithms are simple tests based on the utilization of the processing resource and normally give a sufficient condition for schedulability, while others calculate worst-case response times for the system activities. In the latter group we can find some that are able to estimate the exact response times and others that only can determine an upper bound for these times (we say these algorithms are pessimistic). The scheduling algorithms over which these analysis techniques can be applied are normally fixed priorities (the tasks are scheduled using an integer value) or Earliest Deadline First, EDF (the tasks are scheduled using their deadlines, the task with the closest deadlines are executed first). The blocking times relative to the use of shared resources are also taken into account to perform the analysis.

Normally, we have automatic tools available to apply the schedulability analysis to a system. This is the case of MAST, a Modeling and Analysis Suite for Real-Time Applications [1], which has as its main goal to provide an open source set of tools that enables engineers developing real-time applications to check the timing behavior of their application, including schedulability analysis for checking hard timing requirements.

1.4 Objectives of this Project

The objective of the project is to apply a methodology of real-time analysis to the robot controller CASEVA [5]. The idea is to integrate the results of the WCET measurement tool with a schedulability tool, in order to evaluate the real-time behaviour of the target platform. The specific objectives can be divided into:

- Learning the techniques to measure times and to perform schedulability analysis
- Setting-up MaRTE and CASEVA to measure times
- Analysing the design of the source code of the controlling software CASEVA
- Measurement of the Application
- Building the MAST model for CASEVA
- Applying the schedulability analysis
- Evaluating the overall real-time methodology

Chapter 2

Platform and Tools

This chapter provides a description of the tools and platforms used, and describes the necessary installation steps. However, the preparation of the application code for the measurement itself is part of chapter 3.

2.1 Configuration of Host and Target

As cross compiler platform, and as host for Rapitime, eclipse and MAST, a PC running under Gentoo Linux is used. For the Software transfer over the network device to the target a dhcp server with the following configuration is installed.

/etc/dhcp/dhcpd.conf

```
deny unknown-clients;
always-reply-rfc1048 on;
ddns-update-style none;
subnet 192.168.68.0 netmask 255.255.255.0 {
  ###########
  # caseva  #
  host Target05 {
    hardware ethernet 00:02:B3:B8:38:AB; #MAC of the target
    fixed-address 192.168.68.7;
    next-server 192.168.68.1;
    filename "/home/caseva/export/mprogram";
    option root-path "/home/caseva/export";
  }
}
```

Next, the application emphlinux_eth_receive_log.c has to be compiled for the host PC. It is later used to receive the traced data.

The hardware target platform CASEVA, is a standard PC with special I/O cards. Owing to the absence of the robot and continuing problems with the driver of the I/O cards, all measurements are made with dummy-drivers. The result is of course not a real proof, but for validating the process and the tools it is sufficient. The target PC has a single processor Intel® Pentium® 4 at 2.8 GHz and with 512 MB RAM. The application of the robot controller is written in Ada language [10].

For the transmission of the software and the measured data, the two PCs are connected with a cross link cable. The attempt to connect them over a switch failed due to receiving wrong measurement data.

The operating system for CASEVA is MaRTE OS 1.9 [3]. MaRTE OS is a Hard Real-Time Operating System for embedded applications that follows the Minimal Real-Time POSIX.13 subset. It provides an easy to use and controlled environment to develop Multi-Thread Real-Time applications. MaRTE OS has been developed in the Computers and Real-Time Group of the University of Cantabria.

The Ada compiler which is finally used is GNAT 2008 [9]. It is installed in a user directory of the cross compiler platform, because the MaRTE OS makes some anticipations of it, and therefore needs writing privileges. The compiler directory is added to the PATH variable and other Ada dependent variables are unset by adding the following lines to *.bashrc*. The unsetting is necessary because otherwise the local Ada version is interfered with by Ada libraries of the system.

/etc/dhcp/dhcpd.conf

```
export PATH=/home/jl/usr/gnat/bin:$PATH
unset ADA_INCLUDE_PATH;
unset ADA_OBJECTS_PATH;
...
```

MaRTE OS is installed via the following commands:

/etc/dhcp/dhcpd.conf

```
./minstall
mkrtsmarteuc -march x86 -mproc pii
mkmarte -march x86 -mproc pii
```

After that, the path of the MaRTE tools, for example, the anticipated compiler *mgnatmake*, is added to the environment varibale PATH. For later compilation it is necessary to pass all library paths which are needed to the MaRTE compiler. Therefore, the bash variable *martelib* is added.

~/.bashrc

```
export PATH=/home/jl/usr/marte/utils:PATH
martelib="\
 -aI/home/jl/usr/marte/arch/hwi/ \
 -aI/home/jl/usr/marte/sll \
 -aO/home/jl/usr/marte/lib \
 -aI/home/jl/usr/marte/kernel \
 -aI/home/jl/usr/marte/lib \
 -aI/home/jl/usr/marte/misc \
 -aI/home/jl/usr/marte/x86_arch/drivers/keyboard_driver/ \
 -aI/home/jl/usr/marte/x86_arch/drivers/ \
 -aI/home/jl/usr/marte/x86_arch/drivers/console_switcher/ \
 -aI/home/jl/usr/marte/x86_arch/drivers/bttv \
 -aI/home/jl/usr/marte/x86_arch/drivers/demo_driver_c \
 -aI/home/jl/usr/marte/x86_arch/drivers/ide \
 -aI/home/jl/usr/marte/x86_arch/drivers/pci_1710 \
 -aI/home/jl/usr/marte/x86_arch/drivers/pcm_3724 \
 -aI/home/jl/usr/marte/x86_arch/drivers/serial_port \
 -aI/home/jl/usr/marte/x86_arch/drivers/can \
 -aI/home/jl/usr/marte/x86_arch/drivers/dynamic_buffer \
 -aI/home/jl/usr/marte/x86_arch/drivers/keyboard_driver \
 -aI/home/jl/usr/marte/x86_arch/drivers/pci_1753 \
 -aI/home/jl/usr/marte/x86_arch/drivers/printer_port \
 -aI/home/jl/usr/marte/x86_arch/drivers/soundblaster \
 -aI/home/jl/usr/marte/x86_arch/drivers/cmps03 \
 -aI/home/jl/usr/marte/x86_arch/drivers/eth_drv \
 -aI/home/jl/usr/marte/x86_arch/drivers/membuffer_driver \
 -aI/home/jl/usr/marte/x86_arch/drivers/pci_6713 \
 -aI/home/jl/usr/marte/x86_arch/drivers/rt61 \
 -aI/home/jl/usr/marte/x86_arch/drivers/svga \
 -aI/home/jl/usr/marte/x86_arch/drivers/console_switcher \
 -aI/home/jl/usr/marte/x86_arch/drivers/fat \
 -aI/home/jl/usr/marte/x86_arch/drivers/mouse \
 -aI/home/jl/usr/marte/x86_arch/drivers/pcl_725 \
 -aI/home/jl/usr/marte/x86_arch/drivers/rt-ep \
 -aI/home/jl/usr/marte/x86_arch/drivers/text_console_driver\
 -aI/home/jl/usr/marte/x86_arch/drivers/demo_driver_ada \
 -aI/home/jl/usr/marte/x86_arch/drivers/i2c \
 -aI/home/jl/usr/marte/x86_arch/drivers/pci \
 -aI/home/jl/usr/marte/x86_arch/drivers/pcm_3718 \
 -aI/home/jl/usr/marte/x86_arch/drivers/sb_3621x"
```

The controlling application CASEVA is written in Ada and mostly obeys the rules of structural programming. It has five concurrent tasks, not counting the main task, which is only an empty loop. It also has 10 shared resources, with about 60 functions to access these resources. The tasks are scheduled by a fixed priority scheduler with the priority ceiling protocol for the shared resources. That ensures that a task, locking a resource, gets the priority from the resource. The priority of the resource is the highest priority of the tasks which are ever using the resource.

For compiling with MaRTE OS, *mgnatmake* is used. All the necessary library paths from the operating system have to be declared, which is why we set the environment variable *martelib*. By copying the main program, *principal* to the export file of the dhcp server (../export/mprogram), the program will be transferred to the target during the next booting process.

/etc/dhcp/dhcpd.conf

```
mgnatmake principal.adb $martelib
```

2.2 Rapitime

Rapitime from Rapita Systems[7] is a commercial tool for measuring the Worst-Case Execution Time of software, written in Ada, C or binary code. The measurement is based on a structural analysis and measurements with the automatic instrumented code. The latest Rapitime version is 2.1d with a debugged adt2ast version.

In the next part you can see an instrumented Ada function. The function *rptinstr-lib.RPT_Ipoint(x)* is added from *adains*. You can see how there are several instrumentation points. For each part, between two points Rapitime determines the worst case of the test runs. With that, it calculates the worst case for each function, which is usually higher than the worst case of an end-to-end measurement. Function calls which are in the analysis scope are also included with their own worst case.

Their are several methods to improve the result. For example it's possible to specify the unrolling of loops. For usual processors with pipelines the first time, the loop needs more time to execute than the future times. And so it would be an overestimation if you use the first time as the worst execution time of the block. You can also define black boxes with a specified WCET, or ignore some blocks, which, for example, are implanted for testing reasons.

You can also specify the scope of the measurement data, which are used for the calculation. For a function which is called inside and outside the measured scope, you can define if Rapitime should use the global WCET or just the WCET of the calls within the scope.

These instructions for instrumentation can either be done in the source code or in a separate file. That can be helpful to create a test-environment for different versions of the same application.

instrumented function Escribe_Fichero

```
------------------------------------------------------------------
-- Escribe en el fichero un mensaje con la excepcion que se
--  ha producido y que corresponde al tipo de error indicado ,
--  y el mensaje de error .
------------------------------------------------------------------
procedure Escribe_Fichero (Tipo_Err: in Alarmas.Tipo_Error;
   Mens_Err: in Alarmas.Mensaje_Error) is
begin
   rptinstrlib.RPT_Ipoint( 64571 );if ESCRITURA_FLASH then
      rptinstrlib.RPT_Ipoint( 10 );case Tipo_Err is
      when Excepcion_Normal=>
         rptinstrlib.RPT_Ipoint( 11 );Put_Line(Fich, "
            Excepcion_Normal :");
      when Excepcion_Grave =>
         rptinstrlib.RPT_Ipoint( 12 );Put_Line(Fich, "
            Excepcion_Grave  :");
      when Excepcion_Fatal =>
         rptinstrlib.RPT_Ipoint( 13 );Put_Line(Fich, "
            Excepcion_Fatal  :");
      end case;rptinstrlib.RPT_Ipoint( 14 );
      Put(Fich, "------>");
      Put(Fich, Mens_Err);
   end if;rptinstrlib.RPT_Ipoint( 64570 );
end Escribe_Fichero;
```

The really big advantage is, if it is working, that the use of that tool is very simple. It consists of the following steps.

1. precompile to generate the *.adt files needed by Rapitime
2. create the instrumented Code with *adains*
3. *do the test run*
4. precompile the test data with traceutils
5. unite structural and measurement information with trace parser
6. calculate WCETs with wcet
7. view report with eclipse or with another database tool

2.3 MAST

Mast (Modeling and Analysis Suite for Real Time) 1.3.7.8 [1] is used to built a model of the application and to integrate the WCET. MAST is an open source set of tools that enables modeling real-time applications and performing timing analysis of those applications. The MAST model can be used in a UML design environment to design real-time applications, representing all the real-time behavior and requirements together with the design information, and allowing an automatic schedulability analysis.

The development of this tool is impulsed, like MaRTE, by the Computers and Real-Time Group of the University of Cantabria. It is a tool to describe and validate real-time systems with different parameters, for example, different scheduling strategies. One part of the tool is to validate if the system is schedulable and to determine the worst response times.

Chapter 3

Timing evaluation of the robot software

3.1 Items to measure

The aim is to determine the WCETs for the 5 periodic tasks and for each function, which is part of a protected type object. The result of that will be the input for MAST. These protected objects are the shared resources. The names of the functions follow the style of Rapitime, adding the line number of its first occurrence if it's necessary.

Task	FUNCTION-Name of the Measured
Light_Manager (GL)	Mandos.Rapi_Gestion_Luces
Trajectory_Planning (PT)	planificador_trayectorias.Rapi_Planificador
Message_Logger (EE)	Principal.Rapi_Escribe_Estado
Reporter (CF)	Reportero.Rapi_Cola_A_Fichero
Servo_Control (CS)	Servos.Rapi_Control

Table 3.1: TASKS of CASEVA

SHARED-Resource	FUNCTION	USED BY
cola_alarmas	Alarmas.Activa:109	PT, CS
	Alarmas.Desactiva:126	PT, CS
	Alarmas.Extrae_Error:142	PT
	Alarmas.Inserta_Error:135	PT, CS, GL

SHARED-Resource	FUNCTION	USED BY
	Alarmas.Lee:96	
	Alarmas.Lee_Todas_Alarmas:101	EE, PT, CS
	Alarmas.Reconoce:117	PT
Gestor_Brazo	Brazo.Actua_Rele:148	PT, CS
	Brazo.Bloquea_En_Estado_Seguro:171	CF
	Brazo.Controla_Servos:159	CS
	Brazo.Desbloquea:181	
	Brazo.Estan_Servos_Habilitables:194	CS
	Brazo.Haz_Servos_Habilitables:189	PT
	Brazo.Inicializa_Posicion:121	CS
	Brazo.Lee_Orientacion:88	EE, PT
	Brazo.Lee_Posicion_Ejes:77	EE, PT, CS
	Brazo.Lee_Posicion_Motores:97	
	Brazo.Lee_Rele:113	
	Brazo.Lee_Servos_Ok:107	EE, PT, CS
cola	Cola_Errores.Escribir	PT
	Cola_Errores.Leer	CF
gestor_Botones	mandos.botones.Detecta_Cambio:98	PT
	mandos.botones.Detecta_Pulsacion:105	PT
	mandos.botones.Lee_Estado_Actual:87	PT
	mandos.botones.Lee_Todos:92	EE, PT
	mandos.botones.Lee_Velocidad:110	EE, PT
Gestor_Fuente	Mandos.fuente.Actua_Salida:381	PT
	Mandos.fuente.Bloquea_En_Estado_Seguro:384	CF
	Mandos.fuente.Desbloquea:386	
	Mandos.fuente.Lee_Altura_AVC:380	EE, PT
	Mandos.fuente.Lee_Entrada:376	PT
	Mandos.fuente.Lee_Oscilacion:379	EE, PT
	Mandos.fuente.Lee_Todas_Entradas:378	EE
	Mandos.fuente.Pon_Bloque_Parametros:383	
Gestor_Luces	Mandos.luces.Apaga:214	PT
	Mandos.luces.Enciende:209	PT
	Mandos.luces.Enciende_Con_Parpadeo:225	PT
	Mandos.luces.Enciende_Con_Parpadeo:231	PT
	Mandos.luces.Enciende_Temporizada:219	PT

SHARED-Resource	FUNCTION	USED BY
	Mandos.luces.Esta_Apagada:239	EE, PT
	Mandos.luces.Finaliza_Prueba_Luces:249	PT
	Mandos.luces.Inicia_Prueba_Luces:244	PT
	Mandos.luces.Temporiza_Luces:433	GL
	Mandos.luces.Visualiza_Familia:319	PT
Gestor_Analogico	monitor_driver_analogico.Lee_Altura_Avc:48	EE, PT
	monitor_driver_analogico.Lee_Oscilacion:41	EE, PT
	monitor_driver_analogico.Lee_Posicion_Eje5:64	EE, PT
	monitor_driver_analogico.Lee_Posicion_Eje6:72	EE, PT, CS
	monitor_driver_analogico.Lee_Velocidad:56	EE, PT
Monitor	pantalla.Error:47	
	pantalla.Escribe:34	
	pantalla.Escribe:35	
	pantalla.Escribe:37	EE
	pantalla.Escribe:40	
	pantalla.Limpia_Pantalla:50	
	pantalla.Warning:44	
Monitor_Articular	planificador_trayectorias.Lee_Consigna_Articular	
	planificador_trayectorias.Pon_Consigna_Articular	PT
Monitor_Lectura	planificador_trayectorias.Lee_Consigna_Posicion	EE
	planificador_trayectorias.Pon_Consigna_Posicion	PT
Datos_Servos	Servos.Escribe_Errores_Posicion:228	CS
	Servos.Inicializa_Servos	PT
	Servos.Lee_Errores_Posicion:57	EE
	Servos.Lee_Nuevo_Punto:208	CS
	Servos.Lee_Pos_Inic:246	CS
	Servos.Nuevo_Punto	PT

3.2 Tuning MaRTE OS for Rapitime

In order to be able to use Rapitime in MaRTE OS, we need to tune it. Therefore, the preparation from Álvaro García Cuesta[6] is used.

driver For storing the measurement data, the driver membuffer is used and to transfer the data to the host PC the Ethernet driver is needed. Therefore, the driver has to be activated, by uncommenting the following lines in *marte/x86_arch/arch_dependent_files/marte-kernel-devices_table.ads*.

file: marte/x86_arch/arch_dependent_files/marte-kernel-devices_table.ads

```
with Ethernet;
with Membuffer_Driver_Import;
   ...

   6 => (Name   => "Ether Drv         ",
         Create => Ethernet.Create_Ac,
         Remove => null,
         Open   => Ethernet.Open_Ac,
         Close  => Ethernet.Close_Ac,
         Read   => Ethernet.Read_Ac,
         Write  => Ethernet.Write_Ac,
         Ioctl  => Ethernet.Ioctl_Ac,
         Delete => null,
         Lseek  => null),

  13 => (Name   => "Circular Buffer ",
         Create => Membuffer_Driver_Import.Create_Ac,
         Remove => Membuffer_Driver_Import.Remove_Ac,
         Open   => null,
         Close  => null,
         Read   => Membuffer_Driver_Import.Read_Ac,
         Write  => Membuffer_Driver_Import.Write_Ac,
         Ioctl  => null,
         Delete => null,
         Lseek  => null),
  ...
   9  => ("/dev/eth0         ", 6, 0, True, Device, False, 0),
   17 => ("/dev/membuff      ", 13, 0, True, Device, False, 0),
```

scheduler If during the execution of one instrumented task another task is activated by the scheduler, Rapitime, by default, cannot differentiate which *IPoints* corresponds to which task. Therefore, the switch between two tasks has to be marked, using a special *IPoint* in the scheduler code.

file: marte/kernel/marte-kernel-scheduler.adb

```
 with Rptinstrlib;

 ...

 ---------------
 -- The context switch is actually carried out in
 --  'Change_To_Context' or 'Context_Switch'.
 ----
 Running_Task := Heir_Task;
 if Old_Task.Magic = TERMINATED then

      -- RapiTime begin
    rptinstrlib.RPT_Context_Switch_To (
    rptinstrlib.RPT_Ipoint_ID_Type (TCB_Ac (Running_Task).Id));
    -- RapiTime end

    -- Here the "old task" leaves the CPU so it
    -- is not necessary to save its context.
    HAL.Change_To_Context
                     (New_Task => Heir_Task.Stack_Ptr'Address);
    Pool_TCBs.Release_TCB (Old_Task);
    -- Release task resources (including stack). It has to be
    -- done after changing the CPU stack register, otherwise
    -- it could produce dynamic memory pool corruption since
    -- local variables would be created on a stack that has
    -- already been freed.
 else
      -- RapiTime begin
    rptinstrlib.RPT_Context_Switch_To (
    rptinstrlib.RPT_Ipoint_ID_Type (TCB_Ac (Running_Task).Id));
    -- RapiTime end
    HAL.Context_Switch (Old_Task => Old_Task.Stack_Ptr'Address,
                        New_Task => Heir_Task.Stack_Ptr'Address
                           );
end if;
```

Furthermore, the libraries for the trace and the data transfer have to be created and copied in the proper directory. These libraries were written by Álvaro García. After that, MaRTE OS has to be rebuilt.

```
~$ gcc -c rpt.c logger.c
~$ cp rpt.o logger.o ../marte/objs/x86_objs/
~$ cp rptinstrlib.ads ../marte/x86_arch/include/
~$ mkmarte -march x86 -mproc pii
```

3.3 Tuning of CASEVA Source Code

The preparation for instrumenting the code is in this case quite simple. The global configuration file instrument.h has only one line, that tells Rapitime to instrument all functions.

file: instrument.h

```
#pragma RPT default_instrument(TRUE);
```

The measurement for the tasks needs some further preparation. The tasks are programmed as a closed loop. On the end of the loop is a delay function which makes the task wait until it is time for the next period. The difficulty is that the task cannot be instrumented and the delay function has to be excluded. Rapitime does not support point-to-point measurements, that is why all the significant code has to be exported to a local function, and the task only calls the function. The name of that function is the name of the task, with the prefix *"rapi_"* and this is the root function which is measured. At that point another problem occurs. For the root function exception handlers are not permitted. That is why the code is exported again to a local function so the actual structure has changed as follows:

original structure

```
task body Control is
...
begin
   -- the code of the task
   delay (Ada.Real_Time.To_Duration (proxima_activacion - Ada.
      Real_Time.Clock ) );
end Control
```

anticipated structure

```
task body Control is
    ...
   procedure Rapi_Control is
        procedure Lets_Go is
        begin
            ...
            -- the code of the task
        end Lets_Go;
   begin
        Lets_Go;
   end Rapi_Control;
begin
```

```
   Rapi_Control;
   delay (Ada.Real_Time.To_Duration (proxima_activacion - Ada.
      Real_Time.Clock ) );
 end Control
```

With the last version of Rapitime, there is no problem to instrument and use the protected functions as root functions. However, the measurements show a much smaller result than the reference measurement. For example the outcome of *Lee_Nuevo_Punto* with Rapitime was $0.064\,\mu s$ and the previous measurement has $1.913\,\mu s$ as result. The explanation is that the calling process of a protected function takes about $2\,\mu s$. The problem is solved, by putting a regular function between the calls and the protected function, and measuring its execution time. Luckily these functions already existed. The now accomplished WCET was $2.177\,\mu s$, which was, as expected, a bit higher than the old end-to-end measurement.

In addition to the source code, there are some functions included for end-to-end measurements. Thus, there are some instructions added to the source code in order to ignore this execution time in the measurement.

```
if MEDICION then
   pragma RPT (Ignore_path);
   Medir_Tiempos.Salva_Tiempo(Ada.Execution_Time.Clock -
      Reloj, id);
 end if;
```

3.4 Taking Measurements

The next step is to generate the *.adt files, instrument the code, compile the program and do the test run to receive the measurement data from the host PC with *linux_eth_receive_log* which has to be run under root privileges.

```
    # generationg of *.adt files
~$ mgnatmake -c -gnatc -gnatt principal.adb $martelib
    # instrumentieren of Code, target dir ../rapitemp
~$ adains --exf ../rapitemp/casva.exf -w rptinstrlib \
        -c ../instrument.h -d ../rapitemp *.adt
    # copy of other necesarry files in directory
    # !! -u !! is important, otherwise
    # the instrumentation will be deleted
~$ cp -u *.ad[bs] ../rapitemp
    # Compile software
```

```
~$ mgnatmake principal.adb  $martelib
    # copy software to export file
~$ cp ../rapitemp/principal /home/caseva/export/mprogram

    # start data collector
~$ sudo ./linux_eth_receive_log
```

The next step is the analysis of the data for the necessary functions. As an example, the analysis of *Alarmas.Activa:109* is shown.

```
xstutils -r Alarmas.Activa:109 *.xsc --scope-filter scope.flt
traceutils --outname-template trace.rpz -f ../little_endian.flt
    -f \
            ../marte.flt -f scope.flt output_file.bin
traceparser power.rtd trace_0.rpz --clock-hz 2_800_000_000 \

           --time-unit microseconds
wcalc servos.rtd
```

The later examination can be done with the Rapitime plug-in for eclipse. Due to the large scale of the function, scripts were written for carrying out all the necessary steps(Appendix C and D).

3.5 Comparison of the Measured Results

The results of the measurements can be seen in the table 3.5 for the tasks and the table 3.5 for the protected functions. The measurement named PtoP, is the reference measurement which is published in Análisis de Tiempo Real del Controlador del Brazo CASEVA[5] and is the result of an end-to-end measurement without Rapitime. The other results are the results of the Rapitime measurement, the Overall Worst Case Execution Time (WOET), the Average Overall Execution Time (AOET), the Minimum Overall Execution Time(MOET) and the Maximum Overall Execution Time (MOET). The appended number is just an index indicating the specific measurement. The MOET corresponds to an end-to-end analysis. The Coverage (cov) indicates how many of the IPoints are used in the test. Usually it should be 100%, but in CASEVA there are several parts which are responsible for other time analyses and other debugging functions, this code is never entered in measurement No. 8, therefore, the coverage of *IPoints* of the controlling part of the software is much better than the number which is shown. Furthermore, not all execution path could be used, because of an unavailable test environment to, for example, raise specific exceptions.

FUNCTION	MOET	PtoP	WOET-8
Mandos.Luces.Rapi_Gestion_Luces	7.440	5.922	57.274
planificador_trayectorias.Rapi_Planificador		133.896	133.896*
Principal.Rapi_Escribe_Estado		9472.431	9472.431*
Reportero.Rapi_Cola_A_Fichero	308.005	14.131	365.476
Servos.Rapi_Control	52.477	55.305	142.739

FUNCTION	AOET-8	BOET-8	cov
Mandos.Luces.Rapi_Gestion_Luces	5.718	5.543	56%
planificador_trayectorias.Rapi_Planificador			
Principal.Rapi_Escribe_Estado			
Reportero.Rapi_Cola_A_Fichero	93.283	9.654	30%
Servos.Rapi_Control	31.172	30.051	49%

Table 3.3: Measurement Results of the Tasks

There are protected functions which are never used in the application, though the execution time of these could not be measured. However, to make the list of protected function complete, they are still in the tables. Some results do not exist in the previous measurements and some could not be achieved because of failures of Rapitime. Although many of them have been solved by Rapita Systems, some of them could not be solved before the publishing of this work. For the later simulation, the necessary values are appraised with the help of some other measurements. They are marked with * in the tables.

SHARED-Resource	FUNCTION	USED BY			
FUNCTION	PtoP	MOET	WOET-8	AOET-8	BOET-8
Alarmas.Activa:109	1.060	1.668	1.807	1.278	0.101
Alarmas.Desactiva:126		1.671	1.801	1.790	0.103
Alarmas.Extrae_Error:142	2.282	3.234	4.991	1983	0.219
Alarmas.Inserta_Error:135		3.502	3.849	2.488	1.016
Alarmas.Lee:96					
Alarmas.Lee_Todas_Alarmas:101	1.603	2.401	2.741	1.311	1073
Alarmas.Reconoce:117		1.325	1.326	1.326	1.210
Brazo.Actua_Rele:148		3.047	3.047	3.047	3.047
Brazo.Bloquea_En_Estado_Seguro:171					
Brazo.Controla_Servos:159	2.953	4.552	11.117	3.314	3.209
Brazo.Desbloquea:181					
Brazo.Estan_Servos_Habilitables:194	1.145	1.784	1.784	1.669	1.553
Brazo.Haz_Servos_Habilitables:189					
Brazo.Inicializa_Posicion:121					
Brazo.Lee_Orientacion:88	7.583	7.007	8.937	5.645	5.250
Brazo.Lee_Posicion_Ejes:77	16.273	26.449	36.489	10.057	9.349
Brazo.Lee_Posicion_Motores:97					
Brazo.Lee_Rele:113					
Brazo.Lee_Servos_Ok:107	3.296	3.411	4.669	2.097	0.450
Cola_Errores.Escribir		0.905	1.000	0.507	0.503
Cola_Errores.Leer					

FUNCTION	PtoP	MOET	WOET-8	AOET-8	BOET-8
mandos.botones.Detecta_Cambio:98	1.749		18.723	2.096	2.003
mandos.botones.Detecta_Pulsacion:105	1.770		3.447	2.046	1.887
mandos.botones.Lee_Estado_Actual:87					
mandos.botones.Lee_Todos:92	6.288		27.100	13.908	12.806
mandos.botones.Lee_Velocidad:110	8.175		7.243	5.697	5.330
Mandos.fuente.Actua_Salida:381	2.661		3.586	2.850	0.844
Mandos.fuente.Bloquea_En_Estado_Seguro:384					
Mandos.fuente.Desbloquea:386					
Mandos.fuente.Lee_Altura_AVC:380	8.403		6.294	5.557	5.040
Mandos.fuente.Lee_Entrada:376	1.828		3.639	2.099	1.863
Mandos.fuente.Lee_Oscilacion:379	7.292		7.959	5.627	6.623
Mandos.fuente.Lee_Todas_Entradas:378	1.828		5.979	3.830	3.429
Mandos.fuente.Pon_Bloque_Parametros:383					
Mandos.luces.Apaga:214	1.810		1.810*		
Mandos.luces.Enciende:209	2.172		3.793	2.884	2.023
Mandos.luces.Enciende_Con_Parpadeo:225					
Mandos.luces.Enciende_Con_Parpadeo:231					
Mandos.luces.Enciende_Temporizada:219					
Mandos.luces.Esta_Apagada:239	1.895		1.650	1.073	1.041
Mandos.luces.Finaliza_Prueba_Luces:249			28.554	13.923	13.923
Mandos.luces.Inicia_Prueba_Luces:244			34.894	13.581	13.581
Mandos.luces.Temporiza_Luces:433	2.962	6.589	55.807	4.846	4.687
Mandos.luces.Visualiza_Familia:319	2.171	3.568	3.956	3.569	3.569

FUNCTION	PtoP	MOET	WOET-8	AOET-8	BOET-8
monitor_driver_analogico.Lee_Altura_Avc:48	2.096		3.511	1.714	1.389
monitor_driver_analogico.Lee_Oscilacion:41	2.623	2.734	2.834	1.864	1.616
monitor_driver_analogico.Lee_Posicion_Eje5:64	1.880	2.292	2.936	1.688	1.279
monitor_driver_analogico.Lee_Posicion_Eje6:72	2.651	5.362	5.577	1.733	1.306
monitor_driver_analogico.Lee_Velocidad:56	2.651	2.747	3.056	1.959	1.784
pantalla.Error:47					
pantalla.Escribe:34					
pantalla.Escribe:35					
pantalla.Escribe:37	549.423	552.195	558.169	185.364	39.484
pantalla.Escribe:40					
pantalla.Limpia_Pantalla:50	549.423	215.478	215.969	214.256	212.767
pantalla.Warning:44		165.341	168.850	146.967	123.653
planificador_trayectorias. Lee_Consigna_Articular					
planificador_trayectorias. Lee_Consigna_Posicion	1.778	2.214	2.444	1.424	1.226
planificador_trayectorias. Pon_Consigna_Articular	2.003	1.661	1.764	1.241	1.179
planificador_trayectorias. Pon_Consigna_Posicion	1.877	2.041	2.373	1.298	1.179
Servos.Escribe_Errores_Posicion:228	1.589	1.934	2.391	1.418	1.277
Servos.Inicializa_Servos					
Servos.Lee_Errores_Posicion:57	2.641	2.149	2.554	1.643	1.239

FUNCTION	PtoP	MOET	WOET-8	AOET-8	BOET-8
Servos.Lee_Nuevo_Punto:208	1.913	2.148	2.177	1.255	1.159
Servos.Lee_Pos_Inic:246	3.927	5.139	5.179	1.471	1.321
Servos.Nuevo_Punto	1.746	1.561	1.774	1.285	1.224

The functions *IPoints*, added from *adains*, have of course there own execution time. And this time is now part of the analysed program. To figure out how much time the execution of those takes, a simple test program was built *ipointmess.adb*. The evaluation shows that their are exactly 144 cycles between two IPoints, which are 0.051 μs. This is, of course, without considering an eventual different processor behavior in a real case, where the *IPoints* are surrounded by another context. An interesting thing about that measurement is that the cycles between the last point and the first point of the loop are exactly 144 cycles. Thus, checking the jump condition and the execution of the jump, needs no time at all. So we can see here one example of the complex behavior of a modern processor. The function *Servos.rapi_control* has used in all executions a maximal 200 *IPoints*. Which means, that the execution time is maximal 10 μs longer than using the uninstructed software. Another example, *brazo.lee_orientation* has a maximal 22 *IPoints*, that means a maximal extension of 1.02 μs. Both assume that no *IPoint* is used more than one time in an execution, which can happen for example with loops or with multiple usage of functions.

file: ipointmess.adb

```
with MaRTE_OS;
with rptinstrlib;
procedure Ipoint_Test is
   pot:Integer := 50;
begin
   for I in 1 .. Pot loop
      rptinstrlib.RPT_Ipoint(16#10#);
      rptinstrlib.RPT_Ipoint(16#11#);
      ...
      rptinstrlib.RPT_Ipoint(16#1E#);
      rptinstrlib.RPT_Ipoint(16#1F#);
   end loop;
   Rptinstrlib.RPT_Output_Trace;
end Ipoint_Test;
```

As the comparison of MOET and PtoP shows, the *IPoints* used should be less. That also seems logical, because *IPoints* are used to separate paths and are often after branches. However, it explains why the MOET is usually a bit higher than the previous point-to-point measurement.

It is remarkable that the calculated WCET in some cases is much higher than the really observed time, for example in Mandos.luces.Temporiza_Luces:433. Thus, that function would be interesting for further investigation and optimisation of the source code. Another curiosity is the time value *pantalla.Limpia_Pantalla:50* of the PtoP measurement, because it's much higher than the WOET. In reality I think it is an error in the previous measurement report, because the PtoP values of *pantalla.Limpia_Pantalla:50* and the *pantalla.Escribe:37* are exactly the same.

It appears, although it is not normal, that there are test data for functions which never should be used. A closer look shows that *pantalla.Limpia_Pantalla:50* is used in Escribe Estado, but only for initialization, so it could be ignored for the regular running. The function *pantalla.Warning:44* is a part of a debugging function, which was activated during the test. For the further calculations these are both ignored.

FUNCTION	Priority	Period	Deadline
Mandos.Luces.Rapi_Gestion_Luces	42	$100\mu s$	$100\mu s$
planificador_trayectorias.Rapi_Planificador	41	$50\mu s$	$50\mu s$
Principal.Rapi_Escribe_Estado	9	$1000\mu s$	$1000\mu s$
Reportero.Rapi_Cola_A_Fichero	30	-	-
Servos.Rapi_Control	42	$5\mu s$	$5\mu s$

Table 3.5: System parameter of tasks

3.6 Schedulability Analysis with MAST

The next part of the investigation is to calculate the Worst-Case Response Time Bound. Therefore, the System has to be modeled with its real-time significant structure and the measurement data. The input file of MAST is a text file with an Ada-like syntax.

The MAST analysis will be discussed with two different sets of input data. One with the previous end-to-end measurements and one with the WOET of the Rapitime measurement. You can find the source code of the input file with the WOET data in appendix A. For the PtoP case the structure is the same, but the time values of the functions change. MAST works with non dimensional time units, in this case they are all declared in μs. The intervals, priorities and deadlines can be seen in table 3.5.

To simplify and avoid failures in building the MAST input file I automated this with another python script. The input file is an Emacs org-mode file with two org-mode tables [11] (table with "|" as separator). Further description of that file is in appendix B.

The first lines *Model_Name* and *Model_Date* are only some informational parameters. After that the system description starts.

Our system CASEVA has one *Prozessing_Resource*, the *Speed_Factor* is set to one (that is the default). Changing of this value can be used to get the behavior for lower or higher clocked processors. The System has one System *Scheduler*, which is running on the *processing_resource*. Furthermore, there are five *Scheduling_Server* which are running under the system *scheduler* which has the same name as the processor (*Processor_1*). Every *Scheduling_Server* has a different priority in the system *Scheduler*.

Furthermore, we have eleven *Shared_Resouces*, which are our protected type objects and many functions of which only one is used (protected function). They are defined as *Operation* and have a parameter *Shared_Resource_List* for mapping to the *Shared_Resource* and an *Worst_Case_Execution_Time*.

	PtoP	WOET
Slack System	6885%	2689%
Slack Servo_Control	8870%	3373%
Slack Trajectory_Planning	36640%	35953%
Slack Light_Manager	$\leq 10^5\%$	$\leq 10^5\%$
Slack Reporter	$\leq 10^5\%$	$\leq 10^5\%$
Slack Message_Logger	$\leq 10^5\%$	$\leq 10^5\%$
WC Response Time Servo_Control	$71.578\mu s$	$179.228\mu s$
WC Response Time Trajectory_Planning	$205.474\mu s$	$332.442\mu s$
WC Response Time Light_Manager	$211.396\mu s$	$370.398\mu s$
WC Response Time Reporter	$225.527\mu s$	$735.874\mu s$
WC Response Message_Logger	unbounded	unbounded

Table 3.6: Results of MAST

Five of the *Operation* have the type *Enclosing* and no *Shared_resource*. These operations correspond to measured task functions. They have an execution time and a list of *Operation* which they use.

The *Transaction* trigger the events of the whole system. In our case they periodically start an enclosed operation, in a scheduling server.

The information whose protected function a task uses is not easy to figure out from the source code, because it must include all direct and all indirect calls of the *root* function. Writing a script for this analysis fails because of the limited time. Therefore, this information is obtained from the earlier investigation[5]. After a consultation we came to the conclusion that all functions, where there is no value in the previous measurement, are only used in an initialization way, and can be ignored for the analysis.

The results of MAST can be seen in Table 3.6. It is obvious that the analyses of Rapitime are much more pessimistic than the previous measurements. However, both of them fulfil the real-time conditions. The slack time represents the level by which the function can increase until the bound of real-time conditions is reached. The three values $\leq 10^5\%$ mark a boundary which is so good that it would not be further calculated.

Chapter 4

Conclusions

4.1 About Rapitime

The concept of the Rapitime tool is a very practical and good one. When it works steadily, a measurement can be made very quickly, and if the instrumentation overhead is not too much, it can also be used in productive environments for further tests with large scales of test data. At the moment the application for the requirements tested with Ada seems to be too buggy for commercial use. On the other hand, the on-line support of Rapitime is very fast. The Rapitime team has solved many bugs in a short period of time. Hopefully this product will be improved in the near future. Furthermore, it would be very helpful if Rapitime could support point-to-point measurements so there would be no need to modify the source code to enclose the task code within a function. It would also be helpful to provide support for exception handlers in the root function. For the purposes of integrating the WCETs of protected operations into the MAST model, it would be helpful that the protected operation could be measured within the calling code. Otherwise, a lot of changes have to be made in the original source code.

The naming of the functions with *xstutils* is a little bit unusual. It is not the fully qualified name, even if it very often happens that the names are the same. The name is put together with the file name (without the extension) where the function is defined and the function name. If this name is used more than once, a colon within the line number is added (file-name.function-name:line-Nr). The line number is selected at the first occurrence in the source code (first checking the *.ads and later the *.adb file). A possibility to list the recognised functions would also be very helpful. The best would be by displaying the full definition of the function like:

```
Pantalla.Escribe:32
```

```
    procedure Escribe (mensaje : in String);
Pantalla.Escribe:34
    procedure Escribe (mensaje : in String;
                       color   : in Console_Management.Colors);
```

The file with the results is a sqlite database file which is very useful for extracting data automatically. A little documentation about that would be good.

logger implementation The logger should have an implementation as a thread for permanent output. With the 500 Mb of Ram available, the measurement time for the actual fully instrumented software is limited to about 25 minutes. For larger tests this might not be enough. When there is a membuffer overrun, even if there is sufficient ram, MaRTE crashes. The transmission of the data only works with a cross-link cable. If it goes over a switch, for uncertain reasons, the data are damaged. To make it useful for large-scale test, for example in an industrial environment, an implementation of a higher level connection, for example IP or other bus systems, would be helpful.

4.2 About MAST

The source file is very large in MAST, even for small models. Changing data for different test runs is inconvenient. On the other hand, with one-file source code, it is really easy to write a script for doing that. The error messages of the compiler are sometimes confusing, as it points to the last line, even if the failure is long before.

4.3 About MaRTE

It is not always very easy to get it running; it is necessary to have the correct version of Gnat and sometimes it conflicts with environment variables of other installed Ada versions, however, MaRTE Os needs only the bin path to the right Ada version, other environment variables can disturb. Therefore, it is best to delete all environment variables which refer to Ada. With MaRTE 1.9, it is necessary to quote all the necessary OS library paths with the compiling command, which is in my opinion a little complicated; using a predefined library variable would be helpful.

4.4 Overall Evaluation of the Real-Time Methodology

One of the objectives proposed in the project was to evaluate the applicability of the methodology for measuring times and analysing real-time systems for a person without any experience in real-time systems, which is my case. Table 4.1 shows the tasks I consider relevant in the development of this project and the times (in hours) spent on each one:

1. Setting up MaRTE OS: installing the environment to work with MaRTE
2. Learning the basis of Rapitime: understanding the measurement process with Rapitime and having a basic experience in its usage.
3. Adapting Rapitime for the target platform: integrating Rapitime with the right versions of the Ada compiler (GNAT) and operating system (MaRTE) used in the CASEVA application.
4. First measurement with Rapitime: tuning the software of the application (CASEVA robot controller) to be able to evaluate times with Rapitime (enclosing code to measure in functions).
5. Measurements of the CASEVA software: taking the complete set of time measurements for CASEVA
6. Solving bugs in Rapitime: detecting and fixing errors in task 3, 4 and 5.
7. Learning the basis of MAST: understanding the MAST model, being able to analyse simple examples
8. Building the MAST model for CASEVA: writing the MAST file with the description of CASEVA based on an existing file of a previous version of the software.
9. Analyzing the schedulability for CASEVA: applying the MAST analysis tools to the model built in task 8 and integrating the execution times obtained in task 5.

4.5 Future Perspectives

For future use it would be helpful to build up a proper user interface which provides an easy way to select and measure many functions. A further step would be a direct connection between Rapitime and MAST to transfer the measurement data for several test runs. A first quite simple way to do that is with *extract_data.py* and *gen_mast_model.py.*

Task	Time (Hours)
Setting up MaRTE OS	50
Learning the basis of Rapitime	30
Adapting Rapitime for the target platform	20
First measurement with Rapitime	30
Measurements of the CASEVA software	15
Solving bugs in Rapitime	100
Learning the basis of MAST	15
Building the MAST model for CASEVA	10
Analyzing the schedulability for CASEVA	5

Table 4.1: Working Time

Maybe even some structural information of the MAST model can be extracted from the Rapitime database, like for example the called protected function of an enclosed function. A further investigation of the execution time by instrumenting MaRTE OS could make the results more precise.

Appendix A

MAST input file

WOET-8 Mast Model

```
Model (
   Model_Name   => Caseva,
   Model_Date   => 2008-02-08);

-- Processing Resources

Processing_Resource (
        Type                        => Regular_Processor,
        Name                        => Processor_1,
        Speed_Factor                     => 1,
        Max_Interrupt_Priority      => 256,
        Min_Interrupt_Priority      => 256);

--
Scheduler
  ( Type                     => Primary_Scheduler,
    Name                     => Processor_1,
    Policy                   =>
      ( Type              => Fixed_Priority,
        Max_Priority      => 255,
        Min_Priority      => 1),
    Host                     => Processor_1);



-- Scheduling Servers

Scheduling_Server (
        Type                                => Fixed_Priority,
        Name                                => Servo_Control,
        Server_Sched_Parameters             => (
                Type                => Fixed_Priority_policy,
                The_Priority              => 42,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server (
        Type                              => Fixed_Priority ,
        Name                              => Trajectory_Planning ,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy ,
                The_Priority              => 41,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server (
        Type                              => Fixed_Priority ,
        Name                              => Light_Manager ,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy ,
                The_Priority              => 40,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server ( -- Cola A Fichero
        Type                              => Fixed_Priority ,
        Name                              => Reporter ,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy ,
                The_Priority              => 30,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);
Scheduling_Server (
        Type                              => Fixed_Priority ,
        Name                              => Message_Logger ,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy ,
                The_Priority              => 9,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

-- Resources

Shared_Resource (
        Type      => Immediate_Ceiling_Resource ,
        Name      => cola_alarmas);

Shared_Resource (
        Type      => Immediate_Ceiling_Resource ,
        Name      => Gestor_Brazo);

Shared_Resource (
        Type      => Immediate_Ceiling_Resource ,
        Name      => cola);
```

```
Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => gestor_Botones);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Fuente);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Luces);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Analogico);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Monitor);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Monitor_Lectura);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Monitor_Articular);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Datos_Servos);



       Operation (
       Type                          => Simple,
       Name                          => Alarmas.Activa_109,
       Worst_Case_Execution_Time     => 1.807,
       Shared_Resources_List         => (cola_alarmas));

       Operation (
       Type                          => Simple,
       Name                          => Alarmas.Extrae_Error_142,
       Worst_Case_Execution_Time     => 4.991,
       Shared_Resources_List         => (cola_alarmas));

       Operation (
       Type                          => Simple,
       Name                          => Alarmas.Lee_Todas_Alarmas_101,
       Worst_Case_Execution_Time     => 2.741,
       Shared_Resources_List         => (cola_alarmas));
```

```
Operation (
Type                        => Simple,
Name                        => Brazo.Bloquea_En_Estado_Seguro_171,
Worst_Case_Execution_Time   => 1,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => Brazo.Controla_Servos_159,
Worst_Case_Execution_Time   => 11.117,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => Brazo.Estan_Servos_Habilitables_194,
Worst_Case_Execution_Time   => 1.784,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => Brazo.Lee_Orientacion_88,
Worst_Case_Execution_Time   => 8.937,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => Brazo.Lee_Posicion_Ejes_77,
Worst_Case_Execution_Time   => 36.489,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => Brazo.Lee_Servos_Ok_107,
Worst_Case_Execution_Time   => 4.669,
Shared_Resources_List       => (Gestor_Brazo));

Operation (
Type                        => Simple,
Name                        => mandos.botones.Detecta_Cambio_98,
Worst_Case_Execution_Time   => 18.723,
Shared_Resources_List       => (gestor_Botones));

Operation (
Type                        => Simple,
Name                        => mandos.botones.Detecta_Pulsacion_105,
Worst_Case_Execution_Time   => 3.447,
Shared_Resources_List       => (gestor_Botones));

Operation (
Type                        => Simple,
Name                        => mandos.botones.Lee_Todos_92,
Worst_Case_Execution_Time   => 27.100,
Shared_Resources_List       => (gestor_Botones));
```

```
Operation (
Type                            => Simple ,
Name                            => mandos . botones . Lee_Velocidad_110 ,
Worst_Case_Execution_Time       => 7.243 ,
Shared_Resources_List           => ( gestor_Botones ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . fuente . Actua_Salida_381 ,
Worst_Case_Execution_Time       => 3.586 ,
Shared_Resources_List           => ( Gestor_Fuente ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . fuente . Lee_Altura_AVC_380 ,
Worst_Case_Execution_Time       => 6.294 ,
Shared_Resources_List           => ( Gestor_Fuente ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . fuente . Lee_Entrada_376 ,
Worst_Case_Execution_Time       => 3.639 ,
Shared_Resources_List           => ( Gestor_Fuente ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . fuente . Lee_Oscilacion_379 ,
Worst_Case_Execution_Time       => 7.959 ,
Shared_Resources_List           => ( Gestor_Fuente ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . fuente . Lee_Todas_Entradas_378 ,
Worst_Case_Execution_Time       => 5.979 ,
Shared_Resources_List           => ( Gestor_Fuente ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . luces . Apaga_214 ,
Worst_Case_Execution_Time       => 1.810 ,
Shared_Resources_List           => ( Gestor_Luces ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . luces . Enciende_209 ,
Worst_Case_Execution_Time       => 3.793 ,
Shared_Resources_List           => ( Gestor_Luces ) ) ;

Operation (
Type                            => Simple ,
Name                            => Mandos . luces . Esta_Apagada_239 ,
Worst_Case_Execution_Time       => 1.650 ,
Shared_Resources_List        => (Gestor_Luces));

Operation (
Type                         => Simple,
Name                         => Mandos.luces.Temporiza_Luces_433 ,
Worst_Case_Execution_Time    => 55.807,
Shared_Resources_List        => (Gestor_Luces));

Operation (
Type                         => Simple,
Name                         => Mandos.luces.Visualiza_Familia_319 ,
Worst_Case_Execution_Time    => 3.956,
Shared_Resources_List        => (Gestor_Luces));

Operation (
Type                         => Simple,
Name                         => monitor_driver_analogico.
    Lee_Altura_Avc_48 ,
Worst_Case_Execution_Time    => 3.511,
Shared_Resources_List        => (Gestor_Analogico));

Operation (
Type                         => Simple,
Name                         => monitor_driver_analogico.
    Lee_Oscilacion_41 ,
Worst_Case_Execution_Time    => 2.834,
Shared_Resources_List        => (Gestor_Analogico));

Operation (
Type                         => Simple,
Name                         => monitor_driver_analogico.
    Lee_Posicion_Eje5_64 ,
Worst_Case_Execution_Time    => 2.936,
Shared_Resources_List        => (Gestor_Analogico));

Operation (
Type                         => Simple,
Name                         => monitor_driver_analogico.
    Lee_Posicion_Eje6_72 ,
Worst_Case_Execution_Time    => 5.577,
Shared_Resources_List        => (Gestor_Analogico));

Operation (
Type                         => Simple,
Name                         => monitor_driver_analogico.
    Lee_Velocidad_56 ,
Worst_Case_Execution_Time    => 3.056,
Shared_Resources_List        => (Gestor_Analogico));

Operation (
Type                         => Simple,
Name                         => pantalla.Escribe_37 ,
Worst_Case_Execution_Time    => 558.169,
Shared_Resources_List        => (Monitor));

Operation (
Type                         => Simple,
Name                         => planificador_trayectorias.
    Lee_Consigna_Posicion,
Worst_Case_Execution_Time    => 2.444,
Shared_Resources_List        => (Monitor_Lectura));

Operation (
Type                         => Simple,
Name                         => planificador_trayectorias.
    Pon_Consigna_Articular,
Worst_Case_Execution_Time    => 1.764,
Shared_Resources_List        => (Monitor_Articular));

Operation (
Type                         => Simple,
Name                         => planificador_trayectorias.
    Pon_Consigna_Posicion,
Worst_Case_Execution_Time    => 2.373,
Shared_Resources_List        => (Monitor_Lectura));

Operation (
Type                         => Simple,
Name                         => Servos.Escribe_Errores_Posicion_228,
Worst_Case_Execution_Time    => 2.391,
Shared_Resources_List        => (Datos_Servos));

Operation (
Type                         => Simple,
Name                         => Servos.Lee_Errores_Posicion_57,
Worst_Case_Execution_Time    => 2.554,
Shared_Resources_List        => (Datos_Servos));

Operation (
Type                         => Simple,
Name                         => Servos.Lee_Nuevo_Punto_208,
Worst_Case_Execution_Time    => 2.177,
Shared_Resources_List        => (Datos_Servos));

Operation (
Type                         => Simple,
Name                         => Servos.Lee_Pos_Inic_246,
Worst_Case_Execution_Time    => 5.179,
Shared_Resources_List        => (Datos_Servos));

Operation (
Type                         => Simple,
Name                         => Servos.Nuevo_Punto,
Worst_Case_Execution_Time    => 1.774,
Shared_Resources_List        => (Datos_Servos));
```

```
 Operation (
 Type     => Enclosing,
 Name     => Light_Manager,
 Worst_Case_Execution_Time  => 57.274,
 Composite_Operation_List   =>
    ( Mandos.luces.Temporiza_Luces_433 ));


 Operation (
 Type     => Enclosing,
 Name     => Trajectory_Planning,
 Worst_Case_Execution_Time  => 133.896,
 Composite_Operation_List   =>
    ( Alarmas.Activa_109,
Alarmas.Extrae_Error_142,
Alarmas.Lee_Todas_Alarmas_101,
Brazo.Lee_Orientacion_88,
Brazo.Lee_Posicion_Ejes_77,
Brazo.Lee_Servos_Ok_107,
mandos.botones.Detecta_Cambio_98,
mandos.botones.Detecta_Pulsacion_105,
mandos.botones.Lee_Todos_92,
mandos.botones.Lee_Velocidad_110,
Mandos.fuente.Actua_Salida_381,
Mandos.fuente.Lee_Altura_AVC_380,
Mandos.fuente.Lee_Entrada_376,
Mandos.fuente.Lee_Oscilacion_379,
Mandos.luces.Apaga_214,
Mandos.luces.Enciende_209,
Mandos.luces.Esta_Apagada_239,
Mandos.luces.Visualiza_Familia_319,
monitor_driver_analogico.Lee_Altura_Avc_48,
monitor_driver_analogico.Lee_Oscilacion_41,
monitor_driver_analogico.Lee_Posicion_Eje5_64,
monitor_driver_analogico.Lee_Posicion_Eje6_72,
monitor_driver_analogico.Lee_Velocidad_56,
planificador_trayectorias.Pon_Consigna_Articular,
planificador_trayectorias.Pon_Consigna_Posicion,
Servos.Nuevo_Punto ));


 Operation (
 Type     => Enclosing,
 Name     => Message_Logger,
 Worst_Case_Execution_Time  => 9472.431,
 Composite_Operation_List   =>
    ( Alarmas.Lee_Todas_Alarmas_101,
Brazo.Lee_Orientacion_88,
Brazo.Lee_Posicion_Ejes_77,
Brazo.Lee_Servos_Ok_107,
mandos.botones.Lee_Todos_92,
mandos.botones.Lee_Velocidad_110,
Mandos.fuente.Lee_Altura_AVC_380,
      Mandos.fuente.Lee_Oscilacion_379 ,
      Mandos.fuente.Lee_Todas_Entradas_378 ,
      Mandos.luces.Esta_Apagada_239 ,
      monitor_driver_analogico.Lee_Altura_Avc_48 ,
      monitor_driver_analogico.Lee_Oscilacion_41 ,
      monitor_driver_analogico.Lee_Posicion_Eje5_64 ,
      monitor_driver_analogico.Lee_Posicion_Eje6_72 ,
      monitor_driver_analogico.Lee_Velocidad_56 ,
      pantalla.Escribe_37 ,
      planificador_trayectorias.Lee_Consigna_Posicion ,
      Servos.Lee_Errores_Posicion_57 ));


       Operation (
       Type     => Enclosing ,
       Name     => Reporter ,
       Worst_Case_Execution_Time  => 365.476,
       Composite_Operation_List   =>
          ( Brazo.Bloquea_En_Estado_Seguro_171 ));


       Operation (
       Type     => Enclosing ,
       Name     => Servo_Control ,
       Worst_Case_Execution_Time  => 142.739,
       Composite_Operation_List   =>
          ( Alarmas.Activa_109 ,
      Alarmas.Lee_Todas_Alarmas_101 ,
      Brazo.Controla_Servos_159 ,
      Brazo.Estan_Servos_Habilitables_194 ,
      Brazo.Lee_Posicion_Ejes_77 ,
      Brazo.Lee_Servos_Ok_107 ,
      monitor_driver_analogico.Lee_Posicion_Eje6_72 ,
      Servos.Escribe_Errores_Posicion_228 ,
      Servos.Lee_Nuevo_Punto_208 ,
      Servos.Lee_Pos_Inic_246 ));


-- Transactions



Transaction (
       Type     => Regular ,
       Name     => Servo_Control ,
       External_Events => (
               (Type            => Periodic ,
                Name            => E1,
                Period          => 5000)),
       Internal_Events => (
               (Type    => regular ,
                name    => O1,
                Timing_Requirements => (
                    Type              => Hard_Global_Deadline ,
                    Deadline          => 5000,
                    referenced_event  => E1))) ,
        Event_Handlers => (
                (Type                   => Activity ,
                 Input_Event            => E1,
                 Output_Event           => O1,
                 Activity_Operation     => Servo_Control ,
                 Activity_Server        => Servo_Control))) ;

Transaction (
        Type     => Regular ,
        Name     => Trajectory_Planning ,
        External_Events => (
                (Type             => Periodic ,
                 Name             => E2,
                 Period           => 50000)) ,
        Internal_Events => (
                (Type    => regular ,
                 name    => O2,
                 Timing_Requirements => (
                    Type              => Hard_Global_Deadline ,
                    Deadline          => 50000,
                    referenced_event  => E2))) ,
        Event_Handlers => (
                (Type                   => Activity ,
                 Input_Event            => E2,
                 Output_Event           => O2,
                 Activity_Operation     => Trajectory_Planning ,
                 Activity_Server        => Trajectory_Planning))) ;

Transaction (
        Type     => Regular ,
        Name     => Light_Manager ,
        External_Events => (
                (Type             => Periodic ,
                 Name             => E3,
                 Period           => 100000)) ,
        Internal_Events => (
                (Type    => regular ,
                 name    => O3,
                 Timing_Requirements => (
                    Type              => Hard_Global_Deadline ,
                    Deadline          => 100000,
                    referenced_event  => E3))) ,
        Event_Handlers => (
                (Type                   => Activity ,
                 Input_Event            => E3,
                 Output_Event           => O3,
                 Activity_Operation     => Light_Manager ,
                 Activity_Server        => Light_Manager))) ;

Transaction (
          Type     => Regular ,
          Name     => Reporter ,
          External_Events => (
                    (Type              => Periodic ,
                     Name              => E4,
                     Period            => 1000000)) ,
          Internal_Events => (
                    (Type     => regular ,
                     name     => O4,
                     Timing_Requirements => (
                              Type                => Hard_Global_Deadline ,
                              Deadline            => 1000000,
                              referenced_event => E4))) ,
          Event_Handlers => (
                    (Type                        => Activity ,
                     Input_Event                 => E4,
                     Output_Event                => O4,
                     Activity_Operation          => Reporter ,
                     Activity_Server             => Reporter))) ;

Transaction (
          Type     => Regular ,
          Name     => Message_Logger ,
          External_Events => (
                    (Type              => Unbounded,
                     Name              => E5,
                     Avg_Interarrival=> 1000000)) ,
          Internal_Events => (
                    (Type     => regular ,
                     name     => O5)) ,
          Event_Handlers => (
                    (Type                        => Activity ,
                     Input_Event                 => E5,
                     Output_Event                => O5,
                     Activity_Operation          => Message_Logger ,
                     Activity_Server             => Message_Logger))) ;
```

Appendix B

gen_mast_model.py

The Python script gen_mast_model.py generates the input file for MAST. It solves the problem of transferring the measurement data from a clear table to the large source code for MAST. Large test scales are therefore no problem at all and there is no possibility of making errors by transferring values.

I chose for the input two Emacs org-mode table. Other tables can be supported by changing the function get_table. The result of that function is just a list of dictionaries, with the indexes of the first line.

The assignment of protected function and enclosing function works over the shortcut (SC) of the protected functions, and the column ENC.

The script generates the enclosing and the protected Operations, the rest of the file is only a fixed string.

For execution it needs the library *orgtable.py*.

The use of the script is:

```
~$ gen_mast_model.py colum_of_meas source-file [target-FILE]
```

The input tables look like this:

table.org

```
* Measurement
** Enclosing
|----+------------+--------+--------+-----|
| SC | FUNCTION     | meas-1 | meas-2 | ... |
|----+--------------+--------+--------+-----|
| GL | Encl_Func1   |  5.543 | ...    |     |
| PT | Encl_Func2   |     32 |        |     |
| EE | ....         |        |        |     |

** Protected
|----------------+---------+-----------------------------+---------|
| SHARED         | ENC     | FUNCTION                    | meas-1  |
|----------------+---------+-----------------------------+---------|
| cola\_alarmas  | PT, CS  | Alarmas.Activa:109          |  1.668  |
| cola\_alarmas  |         | Alarmas.Desactiva:126       |  1.671  |
| cola\_alarmas  | PT      | Alarmas.Extrae\_Error:142   |  3.234  |
```

gen_mast_model.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

__author__ = "Jakob Lombacher <Jakob@Lombacher.net>"
__version__ = "0.1"

mastbegin = """
Model (
   Model_Name  => Caseva,
   Model_Date  => 2008-02-08);

-- Processing Resources

Processing_Resource (
        Type                    => Regular_Processor,
        Name                    => Processor_1,
        Speed_Factor               => 1,
        Max_Interrupt_Priority  => 256,
        Min_Interrupt_Priority  => 256);

--
Scheduler
  ( Type                    => Primary_Scheduler,
    Name                      => Processor_1,
    Policy                    =>
      ( Type              => Fixed_Priority,
        Max_Priority      => 255,
        Min_Priority      => 1),
    Host                      => Processor_1);



-- Scheduling Servers

Scheduling_Server (
        Type                              => Fixed_Priority,
        Name                              => Servo_Control,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy,
                The_Priority              => 42,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server (
        Type                              => Fixed_Priority,
        Name                              => Trajectory_Planning,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy,
                The_Priority              => 41,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server (
        Type                              => Fixed_Priority,
        Name                              => Light_Manager,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy,
                The_Priority              => 40,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);

Scheduling_Server ( -- Cola A Fichero
        Type                              => Fixed_Priority,
        Name                              => Reporter,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy,
                The_Priority              => 30,
                Preassigned               => No),
        Server_Processing_Resource        => Processor_1);
Scheduling_Server (
        Type                              => Fixed_Priority,
        Name                              => Message_Logger,
        Server_Sched_Parameters           => (
                Type              => Fixed_Priority_policy,
                The_Priority              => 9,
                Preassigned               => No),
        Server_Processing_Resource      => Processor_1);

-- Resources

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => cola_alarmas);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Brazo);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => cola);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => gestor_Botones);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Fuente);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Luces);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Gestor_Analogico);

Shared_Resource (
        Type     => Immediate_Ceiling_Resource,
        Name     => Monitor);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Monitor_Lectura);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Monitor_Articular);

Shared_Resource (
     Type        => Immediate_Ceiling_Resource,
     Name        => Datos_Servos);


"""

mastend = """
```

```
-- Transactions



Transaction (
        Type     => Regular,
        Name     => Servo_Control,
        External_Events => (
                (Type           => Periodic,
                 Name           => E1,
                 Period         => 5000)),
        Internal_Events => (
                (Type    => regular,
                 name    => O1,
                 Timing_Requirements => (
                         Type             => Hard_Global_Deadline,
                         Deadline         => 5000,
                         referenced_event => E1))),
        Event_Handlers => (
                (Type                   => Activity,
                 Input_Event            => E1,
                 Output_Event           => O1,
                 Activity_Operation     => Servo_Control,
                 Activity_Server        => Servo_Control)));

Transaction (
        Type     => Regular,
        Name     => Trajectory_Planning,
        External_Events => (
                (Type           => Periodic,
                 Name           => E2,
                 Period         => 50000)),
        Internal_Events => (
                (Type    => regular,
                 name    => O2,
                 Timing_Requirements => (
                         Type             => Hard_Global_Deadline,
                         Deadline         => 50000,
                         referenced_event => E2))),
        Event_Handlers => (
                (Type                   => Activity,
                 Input_Event            => E2,
                 Output_Event           => O2,
                 Activity_Operation     => Trajectory_Planning,
                 Activity_Server        => Trajectory_Planning)));

Transaction (
        Type     => Regular,
        Name     => Light_Manager,
        External_Events => (
                (Type           => Periodic,
                 Name           => E3,
                 Period         => 100000)),
          Internal_Events => (
                    (Type    => regular,
                     name    => O3,
                     Timing_Requirements => (
                              Type             => Hard_Global_Deadline,
                              Deadline         => 100000,
                              referenced_event => E3))),
          Event_Handlers => (
                    (Type                  => Activity,
                     Input_Event           => E3,
                     Output_Event          => O3,
                     Activity_Operation    => Light_Manager,
                     Activity_Server       => Light_Manager)));

Transaction (
          Type      => Regular,
          Name      => Reporter,
          External_Events => (
                    (Type            => Periodic,
                     Name            => E4,
                     Period          => 1000000)),
          Internal_Events => (
                    (Type    => regular,
                     name    => O4,
                     Timing_Requirements => (
                              Type             => Hard_Global_Deadline,
                              Deadline         => 1000000,
                              referenced_event => E4))),
          Event_Handlers => (
                    (Type                  => Activity,
                     Input_Event           => E4,
                     Output_Event          => O4,
                     Activity_Operation    => Reporter,
                     Activity_Server       => Reporter)));

Transaction (
          Type      => Regular,
          Name      => Message_Logger,
          External_Events => (
                    (Type            => Unbounded,
                     Name            => E5,
                     Avg_Interarrival=> 1000000)),
          Internal_Events => (
                    (Type    => regular,
                     name    => O5)),
          Event_Handlers => (
                    (Type                  => Activity,
                     Input_Event           => E5,
                     Output_Event          => O5,
                     Activity_Operation    => Message_Logger,
                     Activity_Server       => Message_Logger)));
```

```
"""

help=r"""
Usage: gen_mast_model name_colum_of_measurement source-file [target-FILE]

gen_mast_model generats the model for a CASEVA like robot,
if the target file is not declared, it use mast_model.txt.
The skript never overides a file.
After the generation it starts the mast generation. And open the output
file in the viewer

The input file is an emacs org mod file of the following
structure. Other content will be ignored.
FUNCTION is the function name SC is a shortcut.
It is used for the comparation which protected function is part of which
Enclosing functions. As name you can use for shure following Letters [A-Za-
    z0-9_-].
The - is substitut to _ for the MAST file. All names are Handeld Case
    sensitive.
Hopefully its a sufficiant anticipation for the MAST syntax.

"
...
* Measurement
** Enclosing
|----+-------------+---------+---------+-----|
| SC | FUNCTION    | meas-1  | meas-2  | ... |
|----+-------------+---------+---------+-----|
| GL | Encl_Func1  |   5.543 | ...     |     |
| PT | Encl_Func2  |      32 |         |     |
| EE | ....        |         |         |     |

** Protected
|---------------+---------+-------------------------------+---------|
| SHARED        | ENC     | FUNCTION                      | meas-1  |
|---------------+---------+-------------------------------+---------|
| cola\_alarmas | PT, CS  | Alarmas.Activa:109            |   1.668 |
| cola\_alarmas |         | Alarmas.Desactiva:126         |   1.671 |
| cola\_alarmas | PT      | Alarmas.Extrae\_Error:142     |   3.234 |
.."

"""


import os, sys, re


class EOF(Exception):
    def __init__(self):
        Exception.__init__(self)

def get_table (infile, path):
    """ Extrakt orgmod Table """
    table = []
    head = []
    try:
        for name in path :
            while True:
                line = infile.readline()
                if len(line) == 0:
                    raise EOF;
                if re.match(r"\*+\W+" + re.escape(name),line ,re.I):
                    print line;
                    break;
        while not re.match(r"\W*\|", line): # search Table
            line = infile.readline()
            if len(line) == 0:
                raise EOF;
        while len(line) != 0:
            if not re.match(r"\w*\|", line):
                break
            elif not re.match(r"\w*\|[-+]*\|", line):
                if head == []:
                    head = [i.strip() for i in re.split(r"\|", line)[1:-1]]
                else:
                    table.append(dict(zip(head, [i.strip() for i in re.
                        split(r"\|", line)[1:-1]])))
            line = infile.readline()
            if len(line) == 0:
                break
    except EOF:
        table=[]
    return table

if len(sys.argv)>2:
    MessName = sys.argv[1]
    orgfilename = sys.argv[2]
else:
    print help
    exit()
if len(sys.argv) > 3:
    mastfilename = sys.argv[3]
else:
    mastfilename = "mast_model.txt"


orgfile = open(orgfilename)
tab_enclosed = get_table(orgfile,["Measurement","Enclosing"])
orgfile.close()


orgfile = open(orgfilename)
tab_protected = get_table(orgfile,["Measurement","protected"])
orgfile.close()
```

```
if not os.access(mastfilename, os.F_OK):
    mastfile = open(mastfilename, "w")
else:
    print "File can not used"
    exit();
    #mastfile = sys.stdout
mastfile.write(mastbegin)

for f in tab_protected:
    if f["ENC"] == "": # if not relevant
        continue
    if f[MessName] == "":
        mess = 1
    else:
        mess = f[MessName]

    mastfile.write("""
       Operation (
       Type                         => Simple,
       Name                         => %s,
       Worst_Case_Execution_Time    => %s,
       Shared_Resources_List        => (%s)); \n"""
                    % (re.sub(r"\\", r"", re.sub(r"\-|:", r"_", f["FUNCTION"
                        ])), mess, re.sub(r"\\", r"", f["SHARED"])))
for e in tab_enclosed:
    if e[MessName] == "":
        mess = 100
    else:
        mess = e[MessName]

    mastfile.write("""
        Operation (
        Type     => Enclosing,
        Name     => %s,
        Worst_Case_Execution_Time   => %s,
        Composite_Operation_List    =>
           ( """
                    % (re.sub(r"\\", r"", re.sub(r"\-|:", r"_", e["NAME"])),
                        re.sub(r"\*", r"", mess)))
    prefix = ""
    for f in tab_protected:
        if re.findall(r"\b" + re.escape(e["SC"]) + r"\b", f["ENC"]):
            mastfile.write( prefix + re.sub(r"\\", r"", re.sub(r"\-|:", r"_
                ", f["FUNCTION"])))
            prefix = ",\n        "
    mastfile.write("  ));\n\n")

mastfile.write(mastend)
mastfile.close()
cmd="mast_analysis parse -v -c -s " + mastfilename + " " + mastfilename + "
    .out "
os.system(cmd)
cmd="mast_analysis classic_rm -c -s " + mastfilename + " " + mastfilename +
      ".out "
os.system(cmd)
cmd="gmastresults " + mastfilename + " " + mastfilename + ".out "
os.system(cmd)
```

Appendix C

inst_code.py

The application *inst_code.py* generates a run-able, instrumented program. Therefore, the source-code will be precompiled, instrumented and compiled. For the instrumentation the file uses the file *instrument.h* which is expected in the source directory. The usage of this application is:

```
~$ inst_code.py marte_path source_dir destination_dir
```

inst_code.py

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

"""
The script is written for MaRTE 1.9

"""

__author__ = "Jakob Lombacher <Jakob@Lombacher.net>"
__version__ = "0.1"
```

```
import os
import sys
import re

martepath = os.path.abspath(sys.argv[1])
sourcepath = os.path.abspath(sys.argv[2])
targetpath = os.path.abspath(sys.argv[3])
workdir = os.getcwd()


# Config
libs_comp=r' -aI' + martepath + r'/arch/hwi/ -aI' + martepath + r'/sll -aO'
     + martepath + r'/lib -aI' + martepath + r'/kernel -aI' + martepath + r
    '/lib -aI' + martepath + r'/misc -aI' + martepath + r'/x86_arch/drivers
    /keyboard_driver/ -aI' + martepath + r'/x86_arch/drivers/ -aI' +
    martepath + r'/x86_arch/drivers/console_switcher/  -aI' + martepath + r
    '/x86_arch/drivers/bttv    -aI' + martepath + r'/x86_arch/drivers/
    demo_driver_c      -aI' + martepath + r'/x86_arch/drivers/ide
                    -aI' + martepath + r'/x86_arch/drivers/pci_1710    -aI' +
     martepath + r'/x86_arch/drivers/pcm_3724        -aI' + martepath + r'/
    x86_arch/drivers/serial_port -aI' + martepath + r'/x86_arch/drivers/can
                     -aI' + martepath + r'/x86_arch/drivers/dynamic_buffer
      -aI' + martepath + r'/x86_arch/drivers/keyboard_driver     -aI' +
    martepath + r'/x86_arch/drivers/pci_1753    -aI' + martepath + r'/
    x86_arch/drivers/printer_port    -aI' + martepath + r'/x86_arch/drivers/
    soundblaster -aI' + martepath + r'/x86_arch/drivers/cmps03
    -aI' + martepath + r'/x86_arch/drivers/eth_drv           -aI' +
    martepath + r'/x86_arch/drivers/membuffer_driver    -aI' + martepath + r
    '/x86_arch/drivers/pci_6713    -aI' + martepath + r'/x86_arch/drivers/
    rt61             -aI' + martepath + r'/x86_arch/drivers/svga -aI' +
    martepath + r'/x86_arch/drivers/console_switcher    -aI' + martepath + r
    '/x86_arch/drivers/fat                  -aI' + martepath + r'/x86_arch/
    drivers/mouse                -aI' + martepath + r'/x86_arch/drivers/
    pcl_725     -aI' + martepath + r'/x86_arch/drivers/rt-ep          -aI' +
     martepath + r'/x86_arch/drivers/text_console_driver -aI' + martepath +
     r'/x86_arch/drivers/demo_driver_ada     -aI' + martepath + r'/x86_arch/
    drivers/i2c                 -aI' + martepath + r'/x86_arch/drivers/pci
                    -aI' + martepath + r'/x86_arch/drivers/pcm_3718    -aI' +
     martepath + r'/x86_arch/drivers/sb_3621x'


if not os.access(targetpath, os.F_OK):
    os.mkdir(targetpath)
else:
    #cleaning
    cmd=r'rm -I -r '+ targetpath + r'/*'
    print cmd
    os.system(cmd)
os.mkdir(targetpath + r"/temp")

# vorbereitung fuer instrumentierung
os.chdir(sourcepath)
```

```
dateien=os.listdir('.')
print "beginn generation adt"
print dateien
for datei in dateien:
    if re.match(r'^.*\.adb$',datei):
        cmd = 'mgnatmake -c -gnatc -gnatt ' + datei + libs_comp
        if os.system(cmd):
            raise Exception;
        print "datei: %s \n %s" % (datei, cmd)
cmd='rm --interactive=once *.ali'
print "%s" % cmd
os.system(cmd)
# instrumentieren

dateien=os.listdir('.')
os.system("pwd")
for datei in dateien:
    if re.match(r'^.*\.adt$',datei):
        cmd = r'adains --exf ' + targetpath + r'/casva.exf -w \'rptinstrlib
            \' -c instrument.h -d ' + targetpath + r'/temp ' +datei
        print "datei: %s \n %s" % (datei, cmd)
        os.system(cmd)
    else:
        print "file %s ignored" % datei
os.chdir(targetpath)
cmd='cp ' + sourcepath + '/* .'
os.system(cmd)
cmd='mv temp/* .'
os.system(cmd)
#cmd='rm mandos-botones.xsc mandos-fuente.xsc mandos-luces.xsc
    configuracion-lee_configuracion.xsc'
#os.system(cmd)
cmd='mgnatmake principal.adb '+ libs_comp
os.system(cmd)
#cmd='cp principal /home/caseva/export/mprogram'
#os.system(cmd)
```

Appendix D

calc_et_data.py

A simple way of calculating all the necessary output data. This script reads the same output data from the org-mode table which is described in the appendix B. Therefore, it is necessary to use the Rapitime names for the function names. It only creates the output and does not export the data in the table, but this can be done with the script *extract_data.py*.

The usage of this application is:

```
~$ calc_et_data.py source_file dir_with_test_data
```

The test data file must include the output.bin with the test data. The parent directory the filter little_endian.flt and marte.flt. The script copies all calculated data in the directory OUT_*function-name* if it succeeds and if the calculation fails in the directory FAIL_OUT_*function-name*. Furthermore, all resulting databases are copied in the directory results, so after the test all OUT_* directories can be deleted. If there was a calculation run before all FAIL_* and OUT_* directories must be deleted before starting again.

calc_et_data.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

__author__ = "Jakob Lombacher <Jakob@Lombacher.net>"
__version__ = "0.2"



import os
import sys
import re
import orgtable


orgfilename = sys.argv[1]
wdir = sys.argv[2]
fail = []
succeed = []

orgfile = open(orgfilename)
list = orgtable.get_table(orgfile,["Measurement","Enclosing"])
orgfile.close()

orgfile = open(orgfilename)
list += orgtable.get_table(orgfile,["Measurement","protected"])
orgfile.close()

os.chdir(wdir);

if not os.access("result", os.F_OK):
    os.mkdir("result")
for I in list:
    status = ""
    if not os.access("OUT_" + I["FUNCTION"], os.F_OK):
        cmd = "xstutils -r " + I["FUNCTION"] + " -o medida.rtd --uniq-
            transition *.xsc --scope-filter scope.flt"
        print(cmd)
        if os.system(cmd):
            status = "FAIL_XST_"
        cmd = "traceutils --outname-template trace.rpz -f ../little_endian.
            flt -f ../marte.flt -f scope.flt output_file.bin"
        print(cmd)
        os.system(cmd)
        files = os.listdir('.')
        for trf in [b for b in files if b.endswith(".rpz") ]:
            cmd = "traceparser medida.rtd " + trf + " --merge-results --
                clock-hz 2_800_000_000 --time-unit microseconds"
            print(cmd)
            if os.system(cmd):
                status = "FAIL_"
        if status == "":
            succeed.append(I["FUNCTION"])
        else:
            fail.append(I["FUNCTION"])
        cmd = "wcalc medida.rtd"
        print(cmd)
        os.system(cmd)
        os.mkdir(status + "OUT_" + I["FUNCTION"])
        if status == "":
            cmd = "cp medida.rtd result/" + I["FUNCTION"] + ".rtd"
            print(cmd)
            os.system(cmd)
        cmd = "mv scope.flt *.rpz medida.rtd " + status + "OUT_" + I["
            FUNCTION"] + "/"
        print(cmd)
        os.system(cmd)
        print("
            ----------------------------------------------------------------------------\
            n")

print "succeed:"
print succeed
print "fail:"
print fail
```

Appendix E

extract_data.py

This tool is to extract data form the Rapitime measurements. Therefore the input and output file is an org-mode table. The package used *orttable.py* is printed in appendix D.

For execution it needs the library *orgtable.py*.

The usage of this application is:

```
~$ calc_et_data.py inout_file.org path_of_rtd_files
   Name_of_Messurement
```

extract_data.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

__author__ = "Jakob Lombacher <Jakob@Lombacher.net>"
__version__ = "0.1"
```

```
import os,sys, sqlite3
import orgtable


orgfilename = sys.argv[1]
path=sys.argv[2]
mess_name=sys.argv[3]
cpu_frequ = "2800.0"                    # in MHz
mess = [[1, "MOET-"],
        [2, "BOET-"],
        [3, "WOET-"]]

output =""
print("All results are in Microseconds")
for orgpath in [["Measurement","Enclosing"], ["Measurement","protected"]]:
    orgfile = open(orgfilename)
    funct = orgtable.get_table(orgfile, orgpath, "hline")
    orgfile.close()
    #output="%s/n%s/n" % (table[0], table[1])
    #print("table:" + table[0]+ " " +table[1] )

    files = os.listdir(path)
    for r in funct:
        if r == "hline":
            continue
        sql = r"""select
  uniqueName  as Function,
  met_max / %s as Max_OverET,
  met_min / %s as Min_OverET,
  (self_wcet + sub_wcet ) / %s as W_OverET
--  (self_met_lroot + sub_met_lroot)/2800.0 as H_OverET
from
  Context inner join Subprogram
    on Context.subprg_id = Subprogram.id
    where uniqueName LIKE "%s"
order by uniqueName;""" % (cpu_frequ,cpu_frequ,cpu_frequ, r["FUNCTION"] )
    #print sql
        print "————————————————————————————————————————"
        for file in files:
            rapidb=sqlite3.connect(path +"/" +file)
            rapidb.row_factory = sqlite3.Row

            c = rapidb.cursor()
            a=c.execute(sql)
            b=a.fetchone()
            if b:
                for m in mess:

                    if not b[m[0]]:
                        "tue nichts"
                    elif m[1] + mess_name not in r:
                        r[m[1]+mess_name]=b[m[0]]
                    elif r[m[1]+mess_name] < b[m[0]]:
```

```
                    print "ersetze %f %f" % (r[m[1]+mess_name], b[m
                        [0]])
                    r[m[1]+mess_name]=b[m[0]]
    rapidb.close


orgtable.emb_table(orgfilename,orgpath, funct)
```

Appendix F

orgtable.py

This is a library to generate and extract org-mode tables.

module: orgtable.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

__author__ = "Jakob Lombacher <Jakob@Lombacher.net>"
__version__ = "0.1"

import os, sys, re


class EOF(Exception):
    def __init__(self):
        Exception.__init__(self)

def get_columns (table):
    key = []
    for r in table:
        if r == "hline":
            continue
        for k in r.keys():
            if not k in key:
                key.append(k)
                key.reverse()
    return key


def make_table (table):
    col = get_columns(table)
    out = "| " +"|".join([str(x) for x in col]) + " | \n"
    for r in table:
        if r == "hline":
            out += "|-\n"
            continue
        out += "| "
        for c in col:
            if c in r.keys():
                out += str(r[c])
            out += " | "
        out += "\n"
    return out;

def emb_table (filename, path, table):
    out = ""
    out_table = make_table(table)
    infile = open(filename)
    try:
        for name in path :
            while True:
                line = infile.readline()
                if len(line) == 0:
                    raise EOF;
                if re.match(r"\*+\W+" + re.escape(name),line ,re.I):
                    out += line
                    break;
                else:
                    out += line
        line = infile.readline()
        while not re.match(r"\W*\|", line): # search Table
            out += line
            line = infile.readline()
            if len(line) == 0:
                raise EOF;
        while len(line) != 0:
            if not re.match(r"\w*\|", line):
                break
            line = infile.readline()
        out += out_table
        while len(line) !=0:
            out += line
            line = infile.readline()
    except EOF:
        table=[]
    infile.close()
    outfile = open(filename, 'w')
    outfile.write(out)
    outfile.close()



def get_table (infile, path, atr=""):
    table = []
    head = []
    try:
        for name in path :
            while True:
                line = infile.readline()
                if len(line) == 0:
                    raise EOF;
                if re.match(r"\*+\W+" + re.escape(name),line ,re.I):
                    print line;
                    break;
        while not re.match(r"\W*\|", line): # search Table
            line = infile.readline()
            if len(line) == 0:
                raise EOF;
        while len(line) != 0: #write tabel
            if not re.match(r"\w*\|", line):
                break
            elif not re.match(r"\w*\|[-+]*\|", line):
                if head == []:
                    head = [i.strip() for i in re.split(r"\|", line)[1:-1]]
                else:
                    table.append(dict(zip(head, [i.strip() for i in re.
                        split(r"\|", line)[1:-1]])))
            elif atr == "hline" and head != []:
                table.append("hline")
            line = infile.readline()
            if len(line) == 0:
                break
    except EOF:
        table=[]
    return table
```

Bibliography

[1] Darke, Harbour, Gutiérrez, Martínez, Medina, Palencia "Modeling and Analysis Suit for Real Time Applications (MAST 1.3.7), http://mast.unican.es/"

[2] M. González Harbour, J.J. Gutiérrez, J.C. Palencia and J.M. Drake. "MAST: Modeling and Analysis Suite for Real-Time Applications". Proceedings of the Euromicro Conference on Real-Time Systems, Delft, The Netherlands, June 2001.

[3] "MaRTE OS 1.9", http://mast.unican.es/

[4] Rapita System Ltd. "RapiTime - User Guide 2.0beta", http://www.rapitasystems.com/

[5] Grupo de Comptadores y Tiempo Real "Análisis de Tiempo Real del Controlador del Brazo CASEVA"

[6] Álvaro García Cuesta "Instrumentación de código para el cálculo de tiempos de ejecución y respuesta en sistemas de tiempo real."

[7] R. Wilhlem, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley,G. Bernat,C. Ferdinand, R. Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Tuschner, J. Staschulat, P. Stenström "The Worst-Case Execution Time Problem - Overview of Methods and Survey of Tools"

[8] J. Liu, "Real-Time Systems" , Prentice Hall, 2000.

[9] AdaCore "GNAT 2008", http://www.adacore.com

[10] S. Tucker Taft, Robert A. Duff, Randall L. Brukardt, Erhard Ploedereder, and Pascal Leroy (Eds.). "Ada 2005 Reference Manual. Language and Standard Libraries. International Standard ISO/IEC 8652:1995(E) with Technical Corrigendum 1 and Amendment 1". LNCS 4348, Springer, 2006.

[11] "Org-Mode", http://orgmode.org/