null
eCos(embedded Configurable operating system)
eCos (embedded Configurable operating system) is one kind of embedded Configurable real-time operating system and suitable for the advanced embedded system application. It is mainly used in consumption electron, telecommunication, vehicle carries system and portable instrument as well as other low costs and portable applications. eCos is also an open source, royalty-free and none copyright limited real-time operating system, applicable for embedded application and development, powerful and full function; full-featured, configurable, expansible, portable and compatible real time embedded kernel; perfectly, it provides royalty and buyout free development tools and widely supports various CPU hardware. Due to these features, eCos becomes the research hotspot of developing new embedded products in the field of new generation embedded operation system, meanwhile it receives more and more embedded amateur’s attention.

This thesis firstly makes a outline about eCos basic characteristic and organizationa structure, then, the main parts of kernel, such as scheduler mechanism, memory management, interrupt and exception handing, are discussed in detail. The detailed comparison between eCos and other open sources, i.e., RT-Linux and uCOS, is performed. The performance are analysed according to the schedule mechanism of multitask, synchronization primitive, real-time characterization. From the comparison, the advantage and weakness of eCos is demonstrated.

Since the schedule mechanism in the real-time operating system is very important, the analysis and optimization of the eCos schedule algorithm with abundant content is implemented in this thesis. The eCos schedler's structure and algorithm principle (including RM EDF) are systematically analyzed. Meanwhile, the platform of eCos software is ported to at ARM9 system with ARM920T kernel (ucdragon S3C2410 development board). The procedure of compile for eCos and its applications is described in detail as well as the debug for device drivers.

Key word: eCos, embedded real-time operating system, component, scheduler
1. Device used to control, monitor, or assist the operation of equipment, machinery or plants

1.1 (real-time embedded operating system, RTOS) / TCP/IP 

1.2
1.3GIS

2

3

4

5

6
1.4 eCos

1.4.1 eCos

1. eCos 1.0 1.1 1.2 1.3 1.4 eCos Cygnus Solutions 1998
2. eCos 1.0 1.1 1.2 1.3 1.4 eCos Cygnus Solutions 1998

RISC | DSP | TCP/IP | IEEE1394 | USB | CAN | Bluetooth | IrDA | Java

Web | HTML | WML

WAP

PDA

GUI

1.4 eCos

1.4.1 eCos
2. 1999 Cygnus EL/IX API eCos EL/IX eCos
3. 1999 11 RedHat 6.74 Cygnus RedHat EL/IX eCos
4. 2000 RedHat eCos 1.3 TCP/IP PCI eCos
5. 2002 RedHat eCos TCP/IP PCI eCos
6. 2003 eCosCentric 2003 5 eCos2.0

1.4.2 eCos

eCos 8 16 32 MPU MCU DSP eCos 8 16 32 HAL HAL eCos 8 16 32 ARM Hitachi H8300 Intel x86 MIPS Matsushita
AM3x Motorola 68k PowerPC SuperH SPARC and NEC V8xx

1.4.3 eCos

RedHat 2004 1 Linux 500KB 1.5MB KB eCos Linux 500KB 1.5MB KB eCos Linux 500KB 1.5MB KB eCos Linux 500KB 1.5MB KB eCos

[1][2][3]
1.5 eCos ARM9 RT Linux uC/OS-II
2.1 eCos

2.1.1 eCos

The majority of configuration options related to interrupt handling are in the HAL packages. Since usually the code has to be platform-specific, there are a number of options provided within the kernel related to slightly higher-level concepts, for example Delayed Service Routines.
2.1 eCos

2.1.2 eCos

1) μITRON3.0 API

2) OSI C

3) ISO C

4) ITRON3.0 API

5) μITRON3.0 API

6) TCP/IP RS-232-USB Bootp/DHCP DNS TFTP/FTP SNMP IPv6 HTTPD PPP
7) JFFS2 Flash  RAM  ROM
8) 
9) GDB debug

2.1.3 eCos

- CVS
- eCos
- eCos


2.2 eCos

- HAL
2.2.1 eCos

2.2 eCos

RedBoot

eCos

Debug

RedBoot

Stub

GDB

[3] eCos

eCos

POSIX

uITRON 3.0

eCos

ARM
API [API] [API] [eCos] [API] [API]
n
```
<table>
<thead>
<tr>
<th>■ eCos</th>
<th>■ eCos</th>
<th>■ TCP/IP</th>
<th>■ OpenBSD</th>
<th>FreeBSD</th>
<th>eCos</th>
<th>FTP</th>
<th>DHTP</th>
<th>SNMP</th>
<th>DNS</th>
<th>HTTPD</th>
<th>IPv6</th>
<th>PPP</th>
<th>eCos</th>
<th>RAM</th>
<th>FLASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td>ROM</td>
<td>ROM</td>
<td>ROM</td>
<td>ROM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANSI C</td>
<td>(uTRON)</td>
<td>POSIX</td>
<td>Web</td>
<td>eCos</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>eCos</td>
<td>(eCos</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(eCos</td>
<td>Configuration Tool)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>eCos</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

2.3 eCos

2.3.1 eCos HAL

- Platform-independent HAL options
- HAL interrupt handling
- HAL context switch support
- Explicit control over cache behaviour
- Source-level debugging support
- ROM monitor support
- Enable use of virtual vector calling interface
  - Inherit console settings from ROM monitor
    - Debug channel is configurable
    - Console channel is configurable
  - Initialize whole of virtual vector table
    - Claim virtual vector table entries
      - Claim virtual vectors
      - Claim version virtual vectors
      - Claim delay virtual vectors
      - Claim cache virtual vectors
      - Claim data virtual vectors
      - Claim non-critical virtual vectors
- 2.3 eCos
2.2.2 eCos

eCos packages [1][4]

(1) packages eCos
docs

(2) tools eCos
docs

(3) examples eCos
docs

(4) doc eCos
docs

eCos (component repository) packages

packages

2.4 packages

(Package Administration Tool) eCos

packages
ecos.db

packages

ecos.db eCos

HAL eCos

eCos eCos

eCos CVS (http://ecos.sourceforge.net/cvsweb.cgi?cvsroot=ecos)
2.4
3.1 eCos

3.1.1 Cygwin

eCos includes support for RedBoot I/O, RedBoot I/O (SMP), and TCP/IP. eCos includes SMP support, which is useful for development.

APIs:

cyg_thread_create cyg_scheduler_lock cyg_mempool_fix_treat

eCos includes support for POSIX and uTRON. eCos also includes support for POSIX pthread_create pthread_mutex_lock
3.1.2

eCos

cyg\pakges\services\memalloc

Cyg_Mempool
Cyg_Mempool2

[1][3]

(variable size memory pool)

(fixed size memory pool)

eCos

MLT

"heap"

"heap"
## 3.1 eCos fallback static array

### 3.1.3 eCos fallback static array

- **CPU**: eCos fallback static array
  - eCos fallback static array
  - VSR
  - VSR
  - eCos
  - VSR
  - eCos
  - VSR
  - eCos
  - VSR
  - eCos
  - VSR
  - eCos
  - VSR
  - eCos
3.1.4 eCos Exception Handing

HAL Exception Support

Platform-independent HAL options

hal_intr.h
3.1.5 eCos

eCos [] [] [] [] (Counter) [] [] (Clock) [] [] (Alarm) [] []
(Timer) [] [] (Counter) [] [] [] [] [] [] [] [] [] [] [] []
Counter [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
tick[] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
Counter [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
Alarm [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
Counter [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
Counter [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
Clock [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []

3.1.6 SMP []

eCos [] [] [] [] (SMP) [] [] [] [] [] [] [] [] [] [] [] []
CPU [] [] [] [] CPU [] [] [] [] [] [] [] [] [] [] [] [] []
cyg_start [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] []
spintlock [] CPU [] []

id [] [] [] [] [] [] [] CPU id [] [] [] [] CPU [] [] [

3.1.7 eCos 3.2.1 RT Linux

CPU id spinlock[0] spinlock[1] CPU id NULL spinlock[0] CPU id 0

SMP eCos API SMP CPU

3.2 eCos

uC/OS-II

3.2.1 RT Linux
3.2.2 uC/OS-II

uC/OS-II is an embedded real-time operating system. It is used in a variety of embedded systems and is compatible with several CPU architectures, including ARM, MIPS, PowerPC, SPARC, Hitachi, and SH. This real-time operating system supports 16-bit and 32-bit microcontrollers, as well as 64-bit processors.

2KBety[5]

uC/OS-II is available in several editions, including uC/OS-1 for 8-bit microcontrollers and uC/OS-2 for 16-bit microcontrollers. It supports various operating modes such as抢占式和非抢占式。

3.2.3 RT Linux

RT Linux is a real-time version of the popular Linux operating system. It includes features that allow it to meet stringent real-time requirements. RT Linux supports multiple CPU architectures, including ARM, MIPS, PowerPC, SPARC, and X86.

RT Linux supports several real-time scheduling algorithms, including EDF and RM. It is designed to provide predictable and deterministic response times for real-time applications.
μ C/OS-II [8] eCos [8] RAM ROM | eCos | 100% 100% [6[7]]

μ C/OS-II eCos | RAM ROM | eCos | 100% 100% [6[7]]}
eCos

4.1 eCos

4.1 eCos

Configure Tool

config/cyg/kernel/mlqueue.hxx
config/cyg/kernel/bitmap.hxx

CYGPRI_KERNEL_SCHED_IMPL_HXX

sched.hxx
4.2.1 Cyg_Scheduleres_Base

eCos C++ packages/kernel/current/src/sched  mlqueue.cxx lottery.cxx bitmap.cxx sched.cxx packages/kernel/current/include Sched.inl mlqueue.hxx lottery.hxx bitmap.hxx sched.hxx

4.2.2 Cyg_Scheduleres_Base

eCos C++ packages/kernel/current/src/sched

class Cyg_Scheduleres_Base

Cyg_Scheduleres_Base

Cyg_Scheduleres_Base
Cyg_Scheduler_Implementation

class Cyg_Scheduler

class Cyg_Scheduler_Implementation

class TCyg_Scheduler_Base

class Cyg_Scheduler

Cyg_Scheduler_Implementation

eCos

class Cyg_SchedThread_Implementation

class Cyg_ThreadQueue_Implementation

eCos sched.cxx

说明：实线表示继承关系；虚线表示友元，箭尾项为箭头项的友元。
CLlist_T<thread>，DNode_T<thread>只有在多级队列调度中才会被继承

4.2
4.2.3 eCos Schedule Bitmap

```
Cyg_sched_bitmap_run_queue //
Cyg_Thread *thread_table[CYGNUM_KERNEL_SCHED_BITMAP_SIZE]
```

Register thread and deregister thread
schedule()

HAL_LSBIT_INDEX(index, run_queue)

index thread_table[2] 2

index CYGNUM_KERNEL_SCHED_BITMAP_SIZE 30

typedef cyg_ucount32 cyges_sched_bitmap

31 30 29 28 ... 3 2 1 0

4.3

0, 29, 30

4.4

typedef Cyg_CList_T<Cyg_Thread> Cyg_RunQueue

Cyg_RunQueue_run_queue[CYGNUM_KERNEL_SCHED_PRIORITIES]

Cyg_CList_T head

Cyg_Sched_bitmap_queue_map

queues_map

head
4.3  (Rate Monotonic, RM)  (Deadline Monotonic, DM)  (Least Laxity First, LLF)  (Earliest Deadline First, EDF)
4.3.1

(RM) (DM)

RM

CPU

U \leq N(\sqrt{2} - 1) \quad (6) [11]

4.3.2

EDF

CPU

LLF
4.3.3 RM

4.3.4.1 eCos
4.5 RM

RM：

1) $3 + 2 = 0$

图示：

线程/任务

RM调度器

优先级映射器

多级队列调度器

位图调度器

内核

4.5 RM
if($x > 0) {
    if(($x > $y) || ($x < $y)) {
        true;
        $max = max($x, $y);
        $min = min($x, $y);
        $y = ($y - 1.3) / ($y - $x);
    }
    else if($x < $y) {
        $x = $y;
        $y = $x;
        if($y != $y) {
            }
    }
    else if($x > $y) {
        $x = $y;
    }
}
RM 4.6
4.6 RM 

RM Configure Tool

RM maxcycle
mincycle
maxcycle
mincycle
pristep

RM

RM [8][12]
4.3.4 EDF

eCos FIFO EDF FIFO EDF FIFO EDF FIFO EDF FIFO EDF FIFO EDF FIFO EDF FIFO

4.7 EDF

EDF = 5 + (0 1 2 3 4) 5 12 30 (31)

4.4 eCos

I/O DEV I/O DEV I/O DEV I/O DEV I/O DEV I/O DEV I/O DEV I/O DEV
4.4.1. eCos

I/O

Kernel

I/O

DEV

HAL

4.8 eCos
4.8 eCos I/O API

(1) eCos I/O API

(2) eCos I/O API kernel

HAL (Hardware Abstraction Layer)

(3) HAL

(4) I/O

(5) DEV

4.4.2 eCos I/O API
1) I/O API

<table>
<thead>
<tr>
<th>I/O API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cyg_io_lookup()</td>
<td>Invokes a callback function to look up an I/O</td>
</tr>
<tr>
<td></td>
<td>device by name or ID.</td>
</tr>
<tr>
<td>cyg_io_read()</td>
<td>Reads data from an I/O device.</td>
</tr>
<tr>
<td>cyg_io_write()</td>
<td>Writes data to an I/O device.</td>
</tr>
<tr>
<td>cyg_io_get_config()</td>
<td>Retrieves configuration settings for</td>
</tr>
<tr>
<td></td>
<td>an I/O device.</td>
</tr>
<tr>
<td>cyg_io_set_config()</td>
<td>Sets configuration settings for an I/O</td>
</tr>
<tr>
<td></td>
<td>device.</td>
</tr>
</tbody>
</table>

2) eCos HAL API

eCos HAL API is an interface between eCos and the hardware. It includes functions for
reading and writing data, as well as setting and getting configuration settings for
I/O devices. This allows for seamless integration between the software and
hardware components of the system.
3) Linux API

Linux API allows Linux developers to create and use a variety of device drivers and I/O interfaces. eCos API provides similar functionality for eCos systems.

```c
struct DevTable Entry
```

1) DEVTAB_ENTRY

```
struct DEVTABLEY {
    struct DEVIO_TABLEY * dev_table;
    struct I/O API * handlers;
    struct DEVTAB_ENTRY * next;
}
```

2) eCos I/O API

```
eCos I/O API provides similar functionality for eCos systems.
```

3) Linux I/O API

```
Linux I/O API allows Linux developers to create and use a variety of device drivers and I/O interfaces.
```

```c
DEVTAB_ENTRY(l, name, dep_name, handlers, init, lookup, priv)
```

- **l**: a pointer to the device node.
- **name**: the name of the device node.
- **dep_name**: the name of the dependent device.
- **handlers**: a pointer to the handlers function.
- **init**: a pointer to the initialization function.
- **lookup**: a pointer to the lookup function.
- **priv**: a pointer to the private data.

The `DEVTAB_ENTRY` structure is used to store information about a device node in the I/O table.
3.8 priv void

(2) DEVIO_TABLE

DEVIOTABLE I/O

DEVIO_TABLE l,write,read,get_config,set_config
Write read get_config set_config

1.0 I/OAPI

write a read a get_config a set_config a

eCos

init lookup write read get_config set_config

4.4.3 eCos

4.8 eCos

4.9 eCos

[3]
4.9 eCos

I/O API

DEVTAB_ENTRY

......

Handlers

......

ISR

DSR

I/O_API

cyg_io_lookup()

cyg_io_lookup()

(2)

(3)

(4)

I/O_API

handlers

ISR

DSR

I/O_API

I/O_API
eCos HAL

5.1 eCos HAL

Architecture HAL

Variant HAL

Platform HAL

ARM

PowerPC

MIPS

CPU

Cache

MMU

FPU

I/O


5.2 HAL

5.2.1

1) 

UART Redboot [1][3]

(1) eCos CDL MLT <arch>_<variant>_<platform> CDL MLT

(2) CDL CYGHWR_MEMORY_ LAYOUT CPU CLD HAL CDL

(3) ecos.db HAL CDL

(4) include/pkgconf MLT .h .ldi

(5) misc/redboots_<STARTUP>.ecm

(6) plt_io.h

(7) Cache MMU

(8) 1/O SDRAM PLL

(9) 1/O SDRAM PLL

(10) ENTRY RedBoot RedBoot

(11) ENTRY RedBoot

(1) GDB GDB GDB kill RedBoot

(2) HAL_STUB_PLATFORM_RESET
package CYGPKG_IO_SERIAL_ARM_SMDK2410 {
    alias              {  "Samsung ARM9/SMDK2410 board serial device drivers"
                              devs_serial_arm_smdk2410 smdk2410_serial_driver }

    hardware
}
directory        devs/serial/arm/smdk2410
script             ser_arm_smdk2410.cdl
description       "Samsung ARM9/SMDK2410 board serial device drivers"

package CYGPKG_HAL_ARM_ARM9_SMDK2410 {
    alias { "Samsung ARM9/SMDK2410 development board"
hal_arm_arm9_smdk2410 }
    directory    hal/arm/arm9/smdk2410
    script       hal_arm_arm9_smdk2410.cdl
    hardware
    description  "The SMDK2410 HAL package provides the support needed to run
eCos on Samsung SMDK2410 and S3c2410x (ARM9) based development
boards."
}

target smdk2410 {
    alias { "Samsung ARM9/SMDK2410 development board" smdk2410 s3c2410x }
    packages {   CYGPKG_HAL_ARM
                     CYGPKG_HAL_ARM_ARM9
                     CYGPKG_HAL_ARM_ARM9_SMDK2410
                     CYGPKG_IO_SERIAL_ARM_SMDK2410
                     CYGPKG_DEVS_FLASH_ARM_SMDK2410
                     CYGPKG_DEVS_FLASH_AMD_AM29XXXXX
     }
    description  "
The SMDK2410 target provides the packages needed to run eCos on Samsung S3c2410x (ARM920T) based development boards (SMDK2410)."

} packages SMDK2410 ARM SMDK2410 CPU Flash

(2) CDL

CDL CDL CYGPKG_HAL_<architecture>_<variant>_<platform> SMDK2410

cdl_package CYGPKG_HAL_ARM_ARM9_SMDK2410 {

display "Samsung ARM9/SMDK2410 development board"

parent CYGPKG_HAL_ARM_ARM9

requires CYGPKG_HAL_ARM_ARM9_ARM920T

hardware

include_dir cyg/hal
define_header hal_arm_arm9_smdk2410.h
description "The SMDK2410 HAL package provides the support needed to run eCos on Samsung S3C2410X (ARM920T) based boards."

compile smdk2410_misc.c hal_diag.c

implements CYGINT_HAL_DEBUG_GDB_STUBS

implements CYGINT_HAL_DEBUG_GDB_STUBS_BREAK

implements CYGINT_HAL_VIRTUAL_VECTOR_SUPPORT

define_proc {
puts $::cdl_system_header "#define CYGBLD_HAL_TARGET_H
<pkgconf/hal_arm.h>"
puts $::cdl_system_header "#define CYGBLD_HAL_VARIANT_H
<pkgconf/hal_arm9.h>"
puts $::cdl_system_header "#define CYGBLD_HAL_PLATFORM_H
<pkgconf/hal_arm9_smdk2410.h>"
puts $::cdl_header "#define HAL_PLATFORM_CPU    "ARM9"
puts $::cdl_header "#define HAL_PLATFORM_BOARD  "SMDK2410 system"
puts $::cdl_header "#define HAL_PLATFORM_EXTRA  ""
puts $::cdl_header "#define HAL_PLATFORM_MACHINE_TYPE  106"
}
puts $::cdl_header "#define HAL_PLATFORM_MACHINE_TYPE  106"

(3) eCos RAM[] ROM[] ROM-RAM[] FLOPPY[] Grub[] CDL[]

cdl_component CYG_HAL_STARTUP {
  display        "Startup type"
  flavor         data
  legal_values   { "RAM" "ROM" "ROMRAM" }
  default_value  { "RAM" }
  no_define
  define -file system.h CYG_HAL_STARTUP
  description    "
When targetting the SMDK2410 evaluation board it is possible to build
the system for either RAM bootstrap or ROM bootstrap(s). Select
'ram' when building programs to load into RAM using eCos GDB stubs.
Select 'rom' when building a stand-alone application which will be
put into ROM, or for the special case of building the eCos GDB stubs
themselves."

```c
no_define
#define
system.h

(4)

CDL

CYGBLD_GLOBAL_OPTIONS
CYGHWR_LAYOUT

eCos

cdl_component CYGPKG_HAL_ARM_ARM9_SMDK2410_OPTIONS {
    display "ARM9/SMDK2410 build options"
    flavor none
    no_define
    description "Package specific build options including control over
    compiler flags used only in building this package,
    and details of which tests are built."

    cdl_option CYGPKG_HAL_ARM_ARM9_SMDK2410_CFLAGS_ADD {
        display "Additional compiler flags"
        flavor data
        no_define
        default_value { "" }
        description ""
This option modifies the set of compiler flags for building the ARM9 SMDK2410 HAL. These flags are used in addition to the set of global flags.

}

cdl_option CYGPKG_HAL_ARM_ARM9_SMDK2410_CFLAGS_REMOVE {
    display "Suppressed compiler flags"
    flavor data
    no_define
    default_value { "" }
    description "This option modifies the set of compiler flags for building the ARM9 SMDK2410 HAL. These flags are removed from the set of global flags if present."

}

dl_component CYGHWR_MEMORY_LAYOUT {
    display "Memory layout"
    flavor data
    no_define
    calculated { CYG_HAL_STARTUP == "RAM" ? "arm_arm9_smdk2410_ram" :
        CYG_HAL_STARTUP == "ROM" ? "arm_arm9_smdk2410_rom" :
        "arm_arm9_smdk2410_romram" }

    "arm_arm9_smdk2410_romram" }

cdl_option CYGHWR_MEMORY_LAYOUT_LDI {
display "Memory layout linker script fragment"
flavor data
no_define
define -file system.h CYGHWR_MEMORY_LAYOUT_LDI
calculated 
   { CYG_HAL_STARTUP === "RAM" ?
      "<pkgconf/mlt_arm_arm9_smdk2410_ram.ldi>" : \ 
      CYG_HAL_STARTUP === "ROM" ?
      "<pkgconf/mlt_arm_arm9_smdk2410_rom.ldi>" : \
      "<pkgconf/mlt_arm_arm9_smdk2410_romram.ldi>" 
   }
}
cdl_option CYGHWR_MEMORY_LAYOUT_H { 
display "Memory layout header file"
flavor data
no_define
define -file system.h CYGHWR_MEMORY_LAYOUT_H
calculated 
   { CYG_HAL_STARTUP === "RAM" ?
      "<pkgconf/mlt_arm_arm9_smdk2410_ram.h>" : \ 
      CYG_HAL_STARTUP === "ROM" ?
      "<pkgconf/mlt_arm_arm9_smdk2410_rom.h>" : \
      "<pkgconf/mlt_arm_arm9_smdk2410_romram.h>" 
   }
}

CDL

RTC ROM Redboot

Redboot

3)
eCos™ 

(1) mlt.h 

(2) ldi.c 
CPU

(3) h.c 
C
RAM
ROM
reserved_

4) CDL

struct channel_data_t

hal_diag.c

50
ISR

```c
xxxxx_ser_channels[] channel_data_t
```

CDL

```c
void cyg_hal_plf_serial_putchar(void * _ch_data, char *c)
```

```c
bool cyg_hal_plf_serial_getc_nonblock(void * _ch_data, cyg_uint8 * ch)
```

```c
int cyg_hal_plf_serial_control(void * _ch_data, _comm_control_cmd_t_func, ...
```

```c
_IOCTL  _COMMCTL_IRQ_ENABLE  _COMMCTL_IRQ_DISABLE
```

```c
int cyg_plf_serial_isr(void * _ch_data, int * _ctrlc, CYG_ADDRWORD_vector, CYG_ADDRWORD_data);
```

GDB Ctrl_C

(1) cyg_hal_plf_serial_getc_nonblock()

(2) cyg_hal_is_break()

(3) true * _ctrlc = 1

(4) CYG_ISR_HANDLED

void cyg_hal_plf_serial_init()

cyg_plf_serial_init_channel()

5.2.2 CPU Cache MMU

CPU CPU CPU CPU CPU

CPU CPU CPU CPU CPU

CPU CPU CPU CPU CPU
1) CPU

eCos
eCos
CPU

Cache

FPU
DSP
MMU

2) CDL

eCos
eCos.db

3) Cache

//Instruction cache

#include (CYGPKG_HAL_ARM_ARM9_ARM920T)

// define HAL_ICACHE_SIZE 0x4000

// define HAL_ICACHE_LINE_SIZE 32

// define HAL_ICACHE_WAYS 64

// define HAL_ICACHE_SETS

(HAL_ICACHE_SIZE/(HAL_ICACHE_LINE_SIZE*HAL_ICACHE_WAYS))

//Data cache

#define HAL_DCACHE_SIZE 0x4000

#define HAL_DCACHE_LINE_SIZE 32

#define HAL_DCACHE_WAYS 64

#define HAL_DCACHE_SETS
(HAL_DCACHE_SIZE/(HAL_DCACHE_LINE_SIZE*HAL_DCACHE_WAYS))

# define HAL_WRITE_BUFFER 64

# define CYGHWR_HAL_ARM_ARM9_CLEAN_DCACHE_INDEX
# define CYGHWR_HAL_ARM_ARM9_CLEAN_DCACHE_INDEX_STEP 0x20
# define CYGHWR_HAL_ARM_ARM9_CLEAN_DCACHE_INDEX_LIMIT 0x100

5.2.3  *

1)  eCos hal arch
2)  HAL CDL
3)  hal_arch.h

(1)  HAL_SaveRegisters() HAL_LSBIT_INDEX() HAL_MSBIT_INDEX() FPU
(2)  HAL_LSBIT_INDEX() HAL_MSBIT_INDEX()
(3)  HAL_THREAD_INIT_CONTEXT() HAL_THREAD_LOAD_CONTEXT() HAL_THREAD_SWITCH_CONTEXT()
(4)  HAL_THREAD_LOAD_CONTEXT() HAL_THREAD_SWITCH_CONTEXT()
(5)  HAL_REORDER_BARRIER()
4) HAL

4) HAL

3. HAL

HAL_BREAKPOINT(label)[]

3. HAL

(6) GDB

HAL_GDB_REGISTERS[]

(7) GDB

HAL_GET_GDB_REGISTERS()

(8) haljmp(buf)

(9) HAL_IDLE_THREAD_ACTION()

(10) CYGNUM_HAL_STACK_SIZE_MINIMUM

(11) Cache

(12) HAL

4) hal_intr.h

1) CPU

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR

VSR
5) VSR

(3) hal_interrupt_handlers[]

(4) Hal_interrupt_data[] hal_interrupt_objects[] hal_vsr_table[]

vectors.S hal_m misc.h

(5) CPU

(6) VSR

5) (1) basetype.h eCos cyg_type.h

(2) hal_io.h I/O

(3) hal_cache.h Cache

(4) arch.inc <architecture>.inc vector.S context.S

6) vector.S CPU FPU
(1) ROM \[arrow\] FLASH
reset[]
_start[]

(2) VSR
VSR[]

(3) CPU
arch.inc
ROM \[arrow\] RAM \[arrow\] BSS

(4) VSR[] VSR[] VSR[]

(5) hal_interrupt_stack_call_pending_dsrs()
hal_arch.h

(6) SP[]

(7) SP[] CPU

(8) cyg_interrupt_call_pending_dsrs()

(9) DSRde[]

(10) CPU

(11) Vectors.S

Vectors.S
5.3 eCos

1)  0x0 reset_vector[]
2)   PLATFORM_SETUP[]
3)   PLATFORM_SETUP ICache, DCache, MMU CPU MMU Cache start[]
4)   start CPU 1/O tick cyg_hal_invoke_constructors()
5)   bss hal_hardware_init()
6)   hal_hardware_init() 1/O tick cyg_hal_invoke_constructors()
7)   cyg_hal_invoke_constructors() C++ ecos LCD IIS
8)   cyg_start()
9)   syg_start() cyg_Scheduler::start_cpu()
6.1  

Nand Flash  Nand Flash  Nor Flash  SDRAM  
Nand Flash  Nand Flash  RedBoot, eCos  BootLoader  
Nand Flash  Nand Flash  Nand Flash  Nand Flash  [13]:  

- BootLoader: 0x00000000-0x00000000  
- RedBoot: 0x00010000-0x00020000  
- eCos: 0x00200000-0x00200000  
- He system: 0x02000000-0x04000000  

- BootLoader  MMU  Cache  BootLoader  
- UART  MMU  Cache  eCos  BootLoader  
- Nand Flash  eCos  eCos  eCos  eCos  eCos  MMU  Cache  MMU  Cache  MMU  Cache  
- RedBoot  BootLoader  RedBoot  SDRAM  RedBoot  RedBoot  RedBoot  
- RedBoot  eCos  MMU  Cache  RedBoot  
- RedBoot  MMU  Cache  RedBoot  (Host)  

59
6.2 eCos

S3C2410  eCos packages/hal/arm/arm9

S3C2410  cdl_component CYGPKG_HAL_ARM_ARM9_S3C2410  ecos.db

S3C2410  cdl_component CYGNUM_HAL_CPUCLOCK  133MHz

ecos.db

C Y G P K G _ H A L _ A R M  3

C Y G P K G _ H A L _ A R M  3

S3C2410  cdl_component CYGPKG_HAL_ARM_ARM9_S3C2410  (configtool)

S3C2410  cdl_component CYGPKG_HAL_ARM_ARM9_S3C2410  (package CYGPKG_HAL_ARM_ARM9_S3C2410)

S3C2410  cdl_option (target S3C2410)

--

60
include/plf_cache.h — cache (17)(18)
include/hal_cache.h
include/hal_platform_ints.h
include/plf_io.h
include/hal_platform_setup.h

src/s3c2510_misc.c — HAL

src/hal_diag.c — eCos printf

misc/redboot_primary_ram.ecm — RAM redboot
misc/redboot_primary_rom.ecm — ROM redboot

platform_setup1 S3C2410 hal_platform_setup.h
platform_setup1 arm9 arm9_misc.c hal_hardware_init
hal_platform_setup.h platform_setup1 vectors.S

S D R A M RAM ROM RAM ROM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM
S D R A M RAM RAM ROM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM RAM

include/pkgconf

R A M ROM ROMRAM 3
6.3 eCos

Configuration Tool Build Templates

6.1 Hardware Samsung ARM9/SMDK2410 development board Tools Paths Build Tool arm-eif-gcc Tools Paths User Tool cygwin bin
6.1 Serial device drivers

Serial device drivers are used to communicate with the hardware. These drivers are provided in the following packages:

- SMDK2410
- SMDK2410 build
- SMDK2410_install
- SMDK2410_mlt

These drivers support the Samsung ARM9/SMDK2410 serial port 1 driver for TTY mode serial device drivers.

6.4 Nand Flash

Nand Flash is used to store the eCos image. The package provided includes:

- Nand Flash
- Nand Flash build
- Nand Flash install
- S32410
- ECC [14]
Int NandReadSinglePage(int blocknum, int page num, char *databuf);
Int NandReadMultiPages(int blocknum, int page num, char *databuf, numofpage);
Int NandWriteSinglePage(int blocknum, int page num, char *databuf);
Int NandWriteMultiPages(int blocknum, int page num, char *databuf, numofpage);
Int NandEraseBlock(int blocknum);
Int NandIsBadblock(int blocknum);
Int NandMarkBadblock(int blocknum);
Int NandReadDeviceID(void);
Int NandInit(void);

Nand Flash 50ns
Flash 16
Flash 38
Nand Flash

6.5 eCos

eCos

1) eCos

2) eCos

3) eCos

64
4) 内存ROM&RAM
5) eCos


Copyright © 2003 by eCosCentric Ltd.


