EVIDENCE®

EMBEDDING TECHNOLOGY

FLEX boards

a quick introduction

…in collaboration with

some details

FLEX

- small size (7x10 cm)
- 100 pin dsPIC
- all pins free on connectors
- 2.54 pitch, no SMD expertise required!
- PIC18 for USB connection

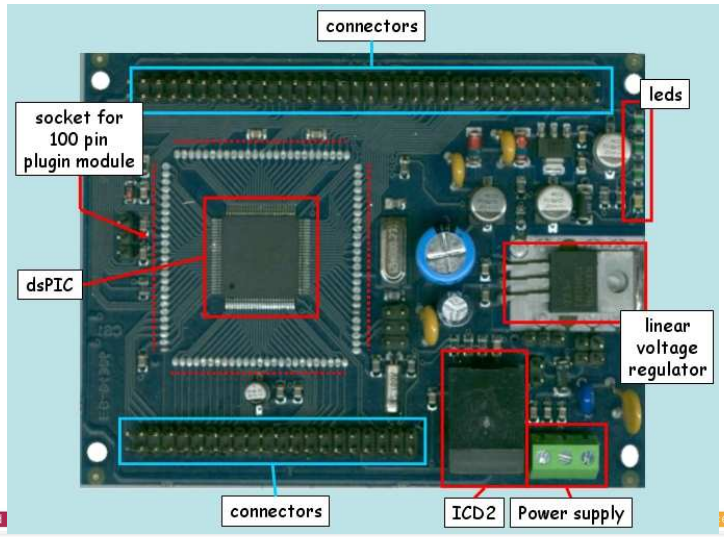
Others

- big!
- limited pin counts MCU
- most of the pins used for LEDs, buttons, …
- difficult to expand!
- no USB

…and also…

- cheap! (99-119 €)
- switching power supply
- resettable fuses
- dsPIC programming from USB (march 2008)
- daughter boards (Thru Hole, CAN, Ethernet, SPI, RS232, RS485, RS422, …)
- software included

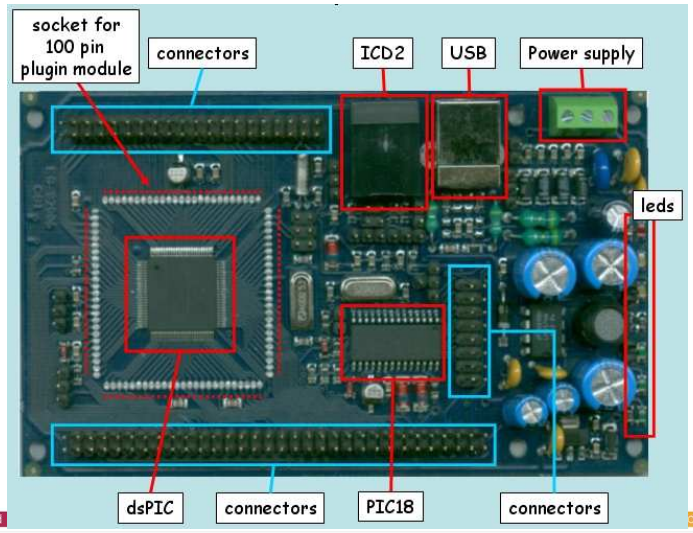
hardware – FLEX Light



5

EVIDENCE EMBEDDING TECHNOLOGY

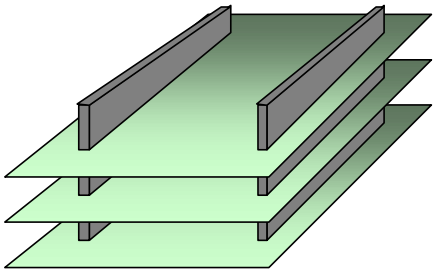
hardware – FLEX Full



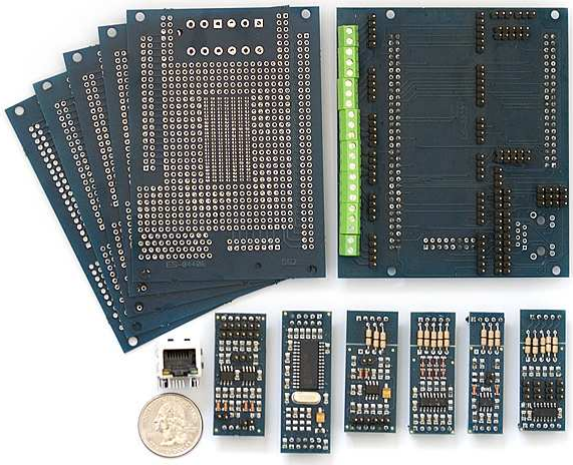
6

EVIDENCE EMBEDDING TECHNOLOGY

board layout



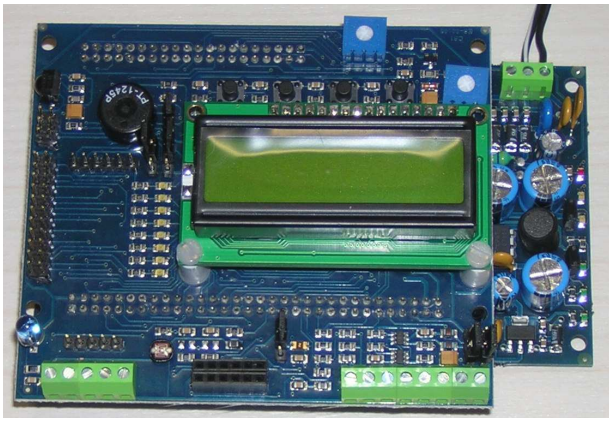
thru hole and multibus board



Available:

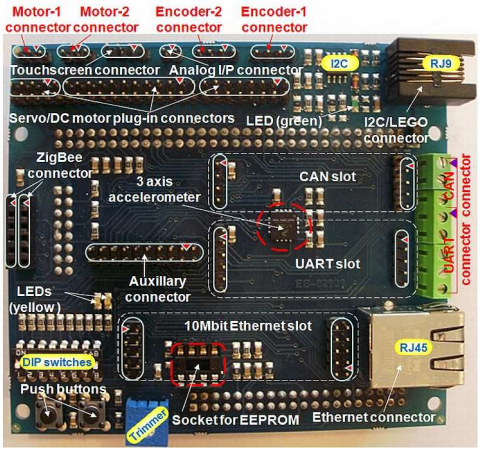
- **Thru Hole**
- **Multibus** (2xCAN, SPI, I2C, Serial, 232, 485, 422, TTL, Ethernet)

demo board



LCD 2x16
8 LED
4 buttons
3-axis Accelerometer
2 DAC
Temperature sensor
Light sensor
Infrared I/O
RS232/485/422 socket
zigbee connector
buzzer
potentiometer
IR in-out
serial I/O

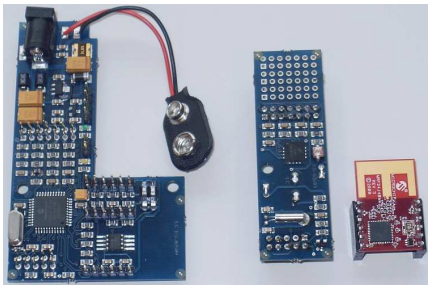
demo2 (motion) board

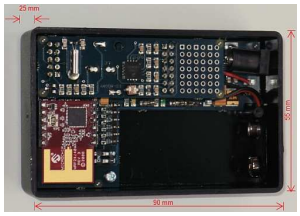


2 DC motors+2Enc plugin
2 Servo+touchscreen plugin

Ethernet
CAN
Serial module
I2C (Lego NXT)
Dip switch
buttons
leds
3-axis accelerometer
Zigbee connector
trimmer
eeprom

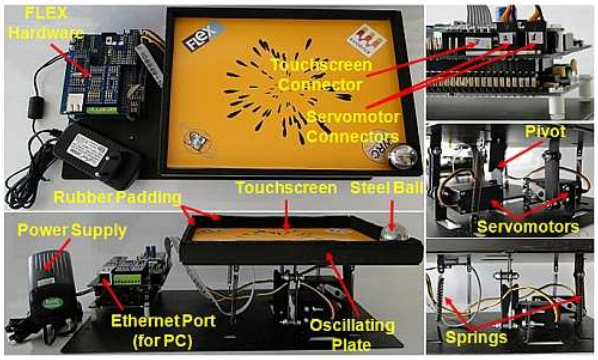
FLEX mini





PIC24FJ64JA004 (16 Mhz)
PIC battery monitor
Real-time clock
1KB Serial EEProm
ZigBee
Pins for PICkit programming
3-axis accelerometer
Buzzer
Thermal sensor
Light sensor
2 x DIP switches
2 x LEDs
9 V battery connector + DC in jack

Amazing Ball



Ball & plate

demo2 board
2 servo motors
touchscreen
power supply

ScicosLab support

ERIKA Enterprise
Configuration details

summary

- ERIKA Enterprise features
- comparison of the various versions
- OIL definition for Microchip dsPIC ® DSC

ERIKA Enterprise

erika enterprise - features

http://erika.tuxfamily.org

supported API
- OSEK OS (BCC1, BCC2, ECC1, ECC2)
- OSEK OIL 1.4.1
- OSEK ORTI 2.1.1 for Lauterbach Trace32

support for
- basic (with stack sharing) / extended tasks
- resources
- events
- hooks
- alarms

erika enterprise

currently available for

- Microchip dsPIC
- ARM7TDMI (Samsung KS32C50100, UniBo MPARM)
- AVR
- Nios II (with multicore support)
- ARM7TDMI (Triscend A7, ST Janus, ST STA2051)
- Tricore 1.3
- PPC e200 z7 Mamba
- PIC32
- Lattice MICO32

and also for

- Hitachi H8 (RCX/Lego Mindstorms)
- C167/ST10 (Ertec EVA 167, tiny/large mem. model)

erika enterprise – licensing and RT-Druid

ERIKA is distributed under the GPL with linking exception license (also known as GNU Crosstool license)

ERIKA Enterprise is available together with the RT-Druid IDE code generator

(will soon be released under the EPL)

- integrated into Eclipse
- code generation for ERIKA Enterprise

comparison

CC	FP, EDF, FRSH
Conformance classes	
• BCC1, BCC2, ECC1, ECC2	• FP (similar to BCC2, or ECC2 if multistack), EDF, FRSH
Startup /Shutdown	
• StartOS, application modes, StartupHook, autostartSystem Shutdown	• No, the main is already the main thread!
• ShutdownOS and ShutdownHook	• No

comparison (2)

Error Handling and Hooks	
• error codes, standard and extended status	• No
• support for ErrorHook and macros	• No
PreTaskHook / PostTaskHook	
• Support for PreTaskHook and PostTaskHook / nothing	• No
ORTI	
• Yes (Nios II)	• Yes (FRSH on Nios II)

comparison (3)

Task	
• TerminateTask and ChainTask	• No (less RAM!)
Informations on tasks	
• GetTaskID and GetTaskState	• No (monostack does not have a task state!)
Basic / extended tasks	
• Basic and Extended Tasks	• blocking primitives to be called within tasks with a private stack

comparison (4)

Number of pending activations	
• BCC1 and ECC1 = only one pending activation. BCC2 and ECC2 = more than one (in OIL file), activations of tasks with same priorities in FIFO order	• the number of pending activations as an integer value, maximum value is implementation dependent. No FIFO order.
Events	
• Yes, in ECC1 and ECC2	• No

comparison (5)

Blocking / non-blocking semaphores

- ECC1/ECC2 Blocking and non blocking semaphores
- BCC1/BCC2 non blocking semaphores
- Semaphore primitives only in multistack configuration.
- Synchronization objects available for FRSH

Primitives for disabling interrupts

- Yes
- No

erika enterprise

- OSEK BCC1, monostack, 2 Tasks, 1 resource, dsPIC

Code footprint (24-bit instructions): 379 (1137 bytes)

- ISR2 stub (for each IRQ) 27
- IRQ end 36
- kernel global functions 99
- ActivateTask 57
- GetResource 12
- ReleaseResource 41
- StartOS 26
- Task end (TerminateTask) 81

Data footprint (bytes)

- ROM 18
- RAM 52

erika enterprise

- FP kernel, monostack, 4 tasks, 1 resource, dsPIC

Code footprint (24-bit instructions):	244 (732 bytes)
• ISR2 stub (for each IRQ)	24
• IRQ end	23
• kernel global functions	67
• ActivateTask	43
• GetResource + ReleaseResource	42
• Task end	45

Data footprint (bytes)

• ROM	26
• RAM	42

board support for dsPIC

ERIKA Enterprise supports the following boards:

- Evidence / Embedded Solutions FLEX board
  supported devices: LEDs, various external devices using add-on boards
- Microchip Explorer 16
  both PIC33 and PIC24
  supported devices: LEDs, Buttons, LCD, Analog
- Microchip dsPICDEM 1.1 Plus
  supported devices: LEDs, Buttons, LCD, Analog, Audio (tbd)

OIL for EE

- the OIL presented in the following slides is a subset of the OSEK OIL standard
- it is a quick tutorial to the OIL definition which can be used for ERIKA Enterprise on the Microchip dsPIC ® DSC
- two columns
  - the first column contains the definition
  - the second column contains examples
- it does not include EDF or FRSH OIL details

OIL (OS object)

definition

```
OIL_VERSION = "2.4";
IMPLEMENTATION ee {
OS {
  STRING EE_OPT[];
  STRING CFLAGS[];
  STRING ASFLAGS[];
  STRING LDFLAGS[];
  STRING LDDEPS[];
  STRING LIBS[];
  BOOLEAN USERESSCHEDULER =
   TRUE;
  […]
```

example

```
CPU mySystem {

OS myOs {
  EE_OPT = "DEBUG";
  EE_OPT = “MYDEFINE”;

  CFLAGS =
   "-IC:/…/scicos";

  USERESSCHEDULER = FALSE;
```

OIL (OS object : CPU data)

definition

```
ENUM [
  […]
  PIC30 {
    STRING APP_SRC[];
    BOOLEAN [
      TRUE {
        BOOLEAN [
          TRUE {
            UINT32 SYS_SIZE;
          },
          FALSE
        ] IRQ_STACK;
      },
      FALSE
    ] MULTI_STACK = FALSE;
    BOOLEAN ICD2 = FALSE;
    BOOLEAN ENABLE_SPLIM = TRUE;
  },
] CPU_DATA[];
```

example

```
CPU_DATA = PIC30 {
  APP_SRC = "code.c";
  MULTI_STACK = FALSE;
  ICD2 = TRUE;
};


CPU_DATA = PIC30 {
  APP_SRC = "code.c";
  MULTI_STACK = TRUE {
    IRQ_STACK = TRUE {
      SYS_SIZE=64;
    };
  };
  ICD2 = TRUE;
  ENABLE_SPLIM = TRUE;
};
```


OIL (OS object : MCU data)

```
ENUM [
  PIC30 {
    ENUM [
      CUSTOM {
        STRING MODEL;
        STRING LINKERSCRIPT;
        STRING DEV_LIB;
        STRING INCLUDE_C;
        STRING INCLUDE_S;
      },
      PIC24FJ128GA006,
      PIC24FJ128GA008,
      […]
    ] MODEL;
  }
] MCU_DATA;
```

```
MCU_DATA = PIC30 {
  MODEL = PIC33FJ256GP710;
};


MCU_DATA = PIC30 {
  MODEL = CUSTOM {
    LINKERSCRIPT =
      "p33FJ256GP710.gld";
    DEV_LIB =
      "libp33FJ256GP710-elf.a";
    INCLUDE_C =
      "p33FJ256GP710.h";
    INCLUDE_S =
      "p33FJ256GP710.inc";
  };
};
```

OIL (OS Object: board data)

```
ENUM [
  NO_BOARD,
  EE_FLEX {
    BOOLEAN USELEDS;
  },
  MICROCHIP_EXPLORER16 {
    BOOLEAN USELEDS;
    BOOLEAN USEBUTTONS;
    BOOLEAN USELCD;
    BOOLEAN USEANALOG;
  }
  MICROCHIP_DSPICDEM11PLUS {
    BOOLEAN USELEDS;
    BOOLEAN USEBUTTONS;
    BOOLEAN USELCD;
    BOOLEAN USEANALOG;
    BOOLEAN USEAUDIO;
  }
  …
] BOARD_DATA = NO_BOARD;
```

```
BOARD_DATA =
  MICROCHIP_EXPLORER16 {
    USELEDS = TRUE;
    USEBUTTONS = TRUE;
    USELCD = TRUE;
    USEANALOG = TRUE;
  };

BOARD_DATA = EE_FLEX {
  USELEDS = TRUE;
};

BOARD_DATA =
      MICROCHIP_DSPICDEM11PLUS {
  USELEDS = TRUE;
  USEBUTTONS = TRUE;
  USELCD = TRUE;
};
```


OIL (OS Object: libraries and kernel type)

```
  ENUM [
    ENABLE {
      STRING NAME;
    }
  ] LIB;

  ENUM [
    FP {
      BOOLEAN NESTED_IRQ;
    },
    BCC1,
    BCC2,
    ECC1,
    ECC2
  ] KERNEL_TYPE;
};
```

```
  LIB = ENABLE {
    NAME = SCICOS;
  };

  KERNEL_TYPE = FP;
};
```

OIL (tasks)

```
TASK {
  UINT32 PRIORITY;
  UINT32 ACTIVATION = 1;
  ENUM [NON, FULL] SCHEDULE;
  TYPE RESOURCE[];
  ENUM [
    SHARED,
    PRIVATE {
      UINT32 SYS_SIZE;
    }
  ] STACK = SHARED;
};
```

```
TASK TaskFlash {
  PRIORITY = 1;
  STACK = SHARED;
  SCHEDULE = FULL;
};

TASK Producer {
  PRIORITY = 2;
  STACK = PRIVATE {
    SYS_SIZE = 64;
  };
  SCHEDULE = FULL;
};
```


OIL (resources)

```
RESOURCE {
  ENUM [
    STANDARD {
      STRING APP_SRC[];
    },
    […]
  ] RESOURCEPROPERTY;
};
```

```
TASK LowTask {
  RESOURCE = "myResource";
  […]
};

RESOURCE myResource {
  RESOURCEPROPERTY=STANDARD;
};
```

OIL (counters and alarms)

```
COUNTER {
  […]
};
ALARM {
  COUNTER_TYPE COUNTER;
  ENUM [
    ACTIVATETASK {
      TASK_TYPE TASK;
    },
    […]
    ALARMCALLBACK {
      STRING
        ALARMCALLBACKNAME;
    }
  ] ACTION;
};
};
```

```
COUNTER myCounter;

ALARM AlarmFlash {
  COUNTER = "myCounter";
  ACTION = ACTIVATETASK {
    TASK = "TaskFlash";
  };
};
```