



Vhodno-izhodne naprave (VIN)

Predavanja

11. Načrtovanje in programiranje vgrajenih sistemov – teorija in praksa

Robert Rozman

rozman@fri.uni-lj.si



Tekoča obvestila

Predstavitel Red Pitaya

VIN projekt – zadnje predavanje



Vsebina

Vsebina I, II:

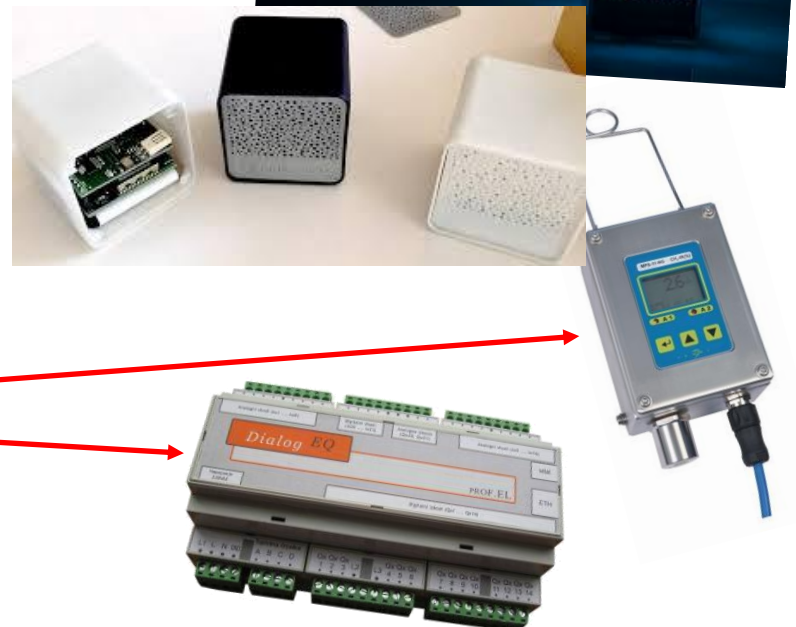
1. Načrtovanje vgrajenih sistemov (HW, SW, ...)
2. Programiranje vgrajenih sistemov – primeri:
 - ▶ Cubesensors, Tevel, D13, Tinia, ...
3. Nivoji programiranja
4. Podrobnejši primeri programiranja – RTOS
 - 4.1 Splošno o RTOS
 - 4.2 FreeRTOS
 - 4.3 MQX RTOS

Vsebina I:

1. Načrtovanje vgrajenih sistemov (HW, SW, ...)

2. Programiranje vgrajenih sistemov :

- ▶ splošen pogled
- ▶ primeri :
 - ▶ A. Cubesensors („pametne kocke“)
 - realizacija v enotni zanki, končni avtomat
 - ▶ B. Tevel – univerzalni merilniki (ekspl. cona)
 - ▶ C. DI3 („pametni hišni regulator“)
 - RTOS (primer MQX)
 - ▶ D. Tinia – Prijazen dom
 - ▶ E. Pametni zabojujnik
 - ▶ F. Embedded Linux (UcLinux, Buildroot)
 - ▶ G. Simulacije
- ▶ CubeIDE: razvoj in razhroščevanje
- ▶ Kaj po razvoju ? Skrb za stabilnost sistemov v praksi
 - ▶ preventiva in kurativa



Vsebina II:

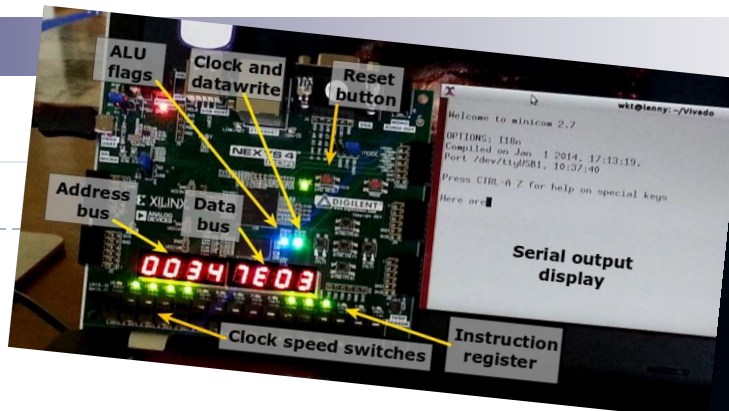
3. Nivoji programiranja

4. Podrobnejši primeri programiranja – RTOS

4.1 Splošno o RTOS

4.2 FreeRTOS

4.3 MQX RTOS



```
void mytask(uint_32 startup_parameter) {  
    /* Task initialization code */  
    ....  
    while (1) {  
        /* Task body */  
        ....  
    }  
}
```

Zakaj HW (in SW) ?

Chipolo - Bluetooth Item Finder for iPhone and Android
by The Chipolo Team

Home Updates **17** Backers **5,329** Comments **1,011**

Funded! This project was successfully funded on November 15, 2013.

5,329 backers
\$293,014
pledged of \$15,000 goal
0 seconds to go

Project by The Chipolo Team
Trbovlje, Slovenia

First created · 0 backed
Has not connected Facebook



CUBESENSORS

Make your home healthier,
your office more productive

For the simple solutions. With just a small, stylish, cordless
connected Cube in each room.

Get Your Cubes Now!

Winter 2013 batch available!



Geoffrey

74844 GUESTS SERVED

826

backers

\$256,125

pledged of \$50,000 goal

0

seconds to go

Funding period

Jul 22, 2013 - Sep 20, 2013 (60 days)



Project by
Red Pitaya
Newport News, VA

**OPEN INSTRUMENTS
FOR EVERYONE**



PLAY



Potato Salad

by Zack Danger Brown

Comments **5,129**

This project was successfully funded on August 2

Columbus, OH

6,911

backers

\$55,492

pledged of \$10 goal

0

seconds to go



Geoffrey



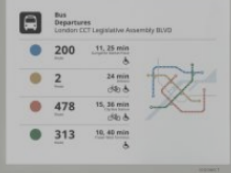
I grant you more
than three wishes.

Drinks

Meals

Specials

STATE-OF-THE-A
CH MAKES THEIR
ME TIME PRESEN
RESTAURANT; CO
SIDER IT AN EXP
ENT IN BETTER BL




**Visionect is a decade-long
market leader in developing
ePaper solutions. Here's why.**

We're constantly testing new technologies and
creating the best solutions or applications of E Ink
systems.

July 20, 2021

is, Thai Inn Pub, I

Zakaj HW (in SW) ?




PRODUKTI IN REŠITVE ZA PAMETNO
AVTOMATIZACIJO ZGRADB

RAZIŠČI

Majhen Koaksialni Helikopter
SCH-2A

Najlažji osebni koaksialni helikopter na svetu...



iSYSTEM ▾ Products & Solutions ▾ Support & Downloads ▾ Partners

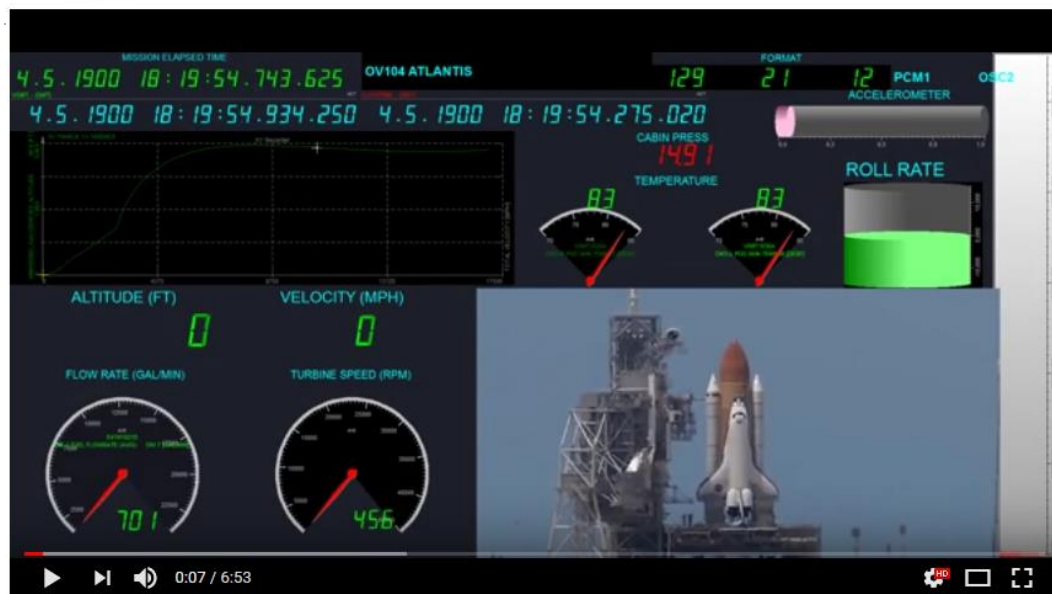
Home > Overview

About

Software	Hardware
winIDEA - IDE, Debug and Trace Tool	BlueBox - Debug & Trace
testIDEA - Software Test Tool	Active Probes
isystem.connect - Automation API	Debug Adapters
daqIDEA - Visualization Tool	Special Adapters
Analyzer - Profiling and Coverage Tool	Emulation Adapters
	Analog/Digital and Network Trace
	Evaluation boards

Past Meetup

Code optimization on modern processors [Dejan Črnica, Dewesoft]



Space shuttle Atlantis launch monitoring with Dewesoft software

Trapview STANDARD

Most widely deployed model of the trap has a similar external appearance as the conventional delta traps, therefore its pest catch efficiency is on the same level with conventional traps as well.



Prikaz primerov vgrajenih sistemov



FRI-SMS



D13 EQ



Tevel
Merilnik konc.
plinov



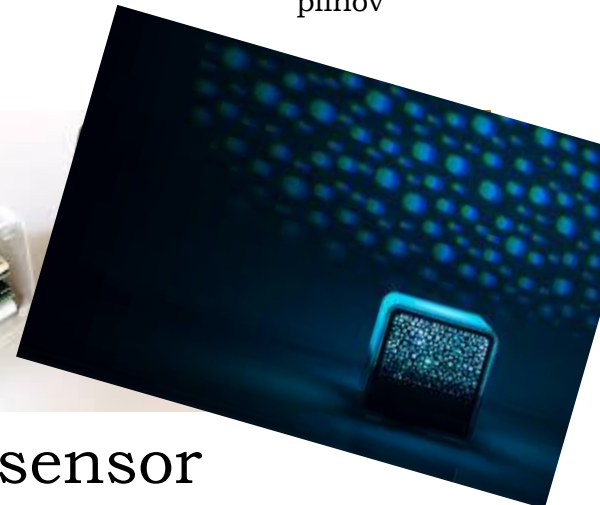
STM Discovery



S1



Cubesensor



1. Načrtovanje vgrajenih sistemov

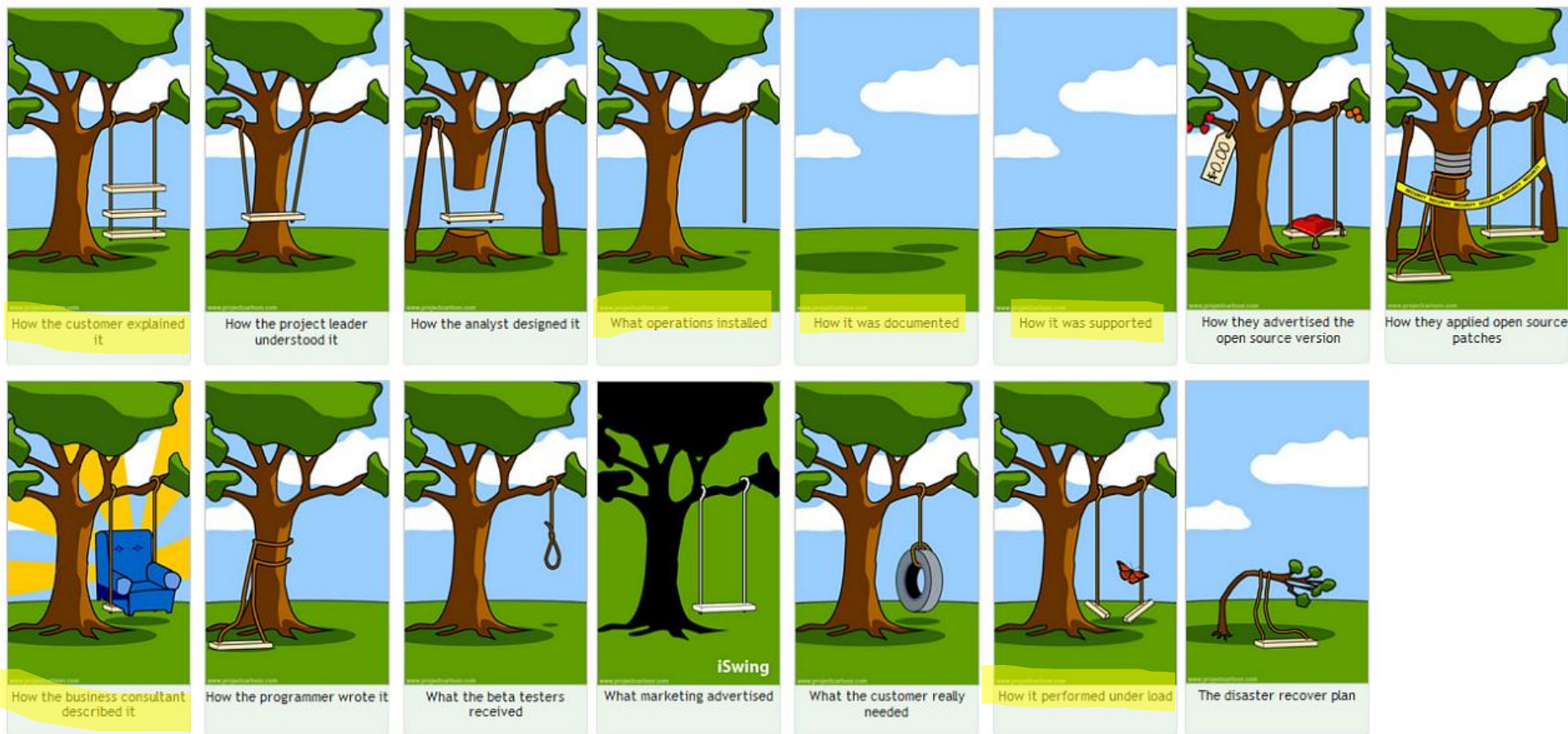
Običajen potek :

- ▶ Specifikacija – opredelitev zahtev ->
 - ▶ zelo pomembna !
- ▶ Izбира el. komponent, čipov, krmilnikov, itd... ->
 - ▶ pregled dokumentacije („Errata“, rok dobavljivosti, podpora,...)
- ▶ Načrtovanje PCB
- ▶ Prvi zagon – oživljanje sistema, razvoj SW
- ▶ Spremljanje delovanja

1. Načrtovanje vgrajenih sistemov

- Specifikacija – opredelitev zahtev
 - zelo pomembna !

Product development from an IT failures perspective



1. Načrtovanje vgrajenih sistemov

Izbira el. komponent, čipov,
krmilnikov, itd...

- ▶ Datasheet (DS):
 - ▶ „kako bi naj delovalo...”
- ▶ Errata:
 - ▶ „kaj vse ne deluje tako kot v DS...”

Prazna ?

Pentium FDIV bug:

The **Pentium FDIV bug** is a **bug** in the **Intel PS Pentium floating point unit (FPU)**. Because of the bug, the processor can return incorrect decimal results, an issue troublesome for the precise calculations needed in fields like math and science.

Errata : + dobro, vsaj znan problem

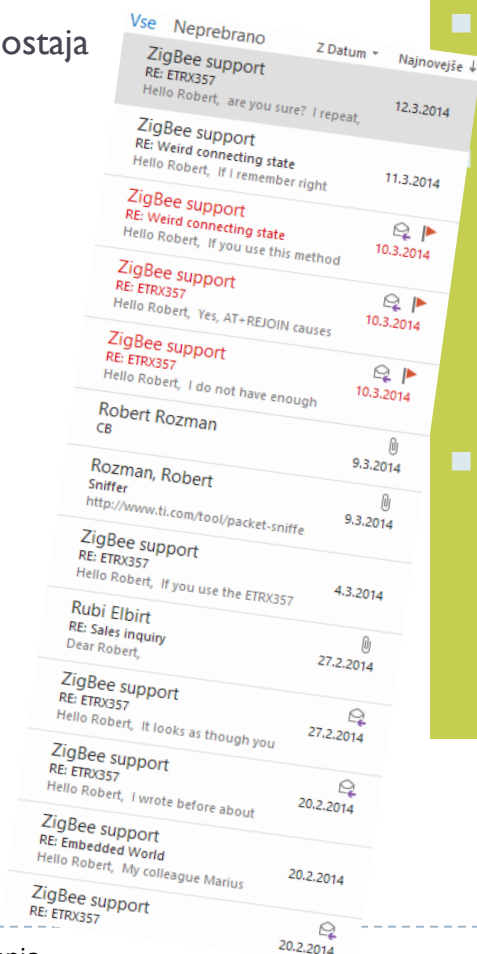
Kaj če naletimo na neznan problem?

- **upamo na reprodukcijo in podporo**

Izbira el. komponent, čipov, krmilnikov, itd... - 2 zgodbi

Podpora - I. zgodba :

- ▶ težave z brezžičnim modulom
 - ▶ cca. 70 emailov,
 - ▶ problem ostaja



Zaznani hrošči:

»stuck in error94«

Main.c : /* **hack when zigbee module gets stuck** in error 94. nothing except AT&F resolves the issue, no ATZ, reset pin, power down */

»no SEQ prompt«

/* Bits are set to 1, when message is in air (SEQ, but not yet ACK or NACK). */
/* **In a perfect world this would not be needed**, but it seems like module
* sometimes (RARELY) does not send SEQ: after AT+UCAST*, but it does send ACK:
* afterwards. If that happens, pending_messages buffer can become -1 long,
* and old (or not yet used, invalid) messages are sent. This is a suspect
* for ticket:185 */

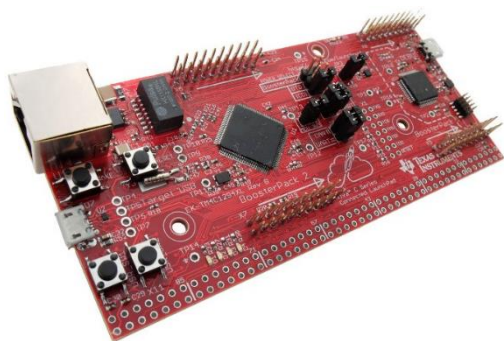
»not be able to reconnect «

```
static void check_initialized(void) {  
    int r;  
    /* zigbee module bug workaround.  
    * Reset the module, or it will not be able to reconnect after seeing  
    * a bunch of modules failing to connect (NEWMODE w/o FFD).  
    */
```


Izbira el. komponent, čipov, krmilnikov, itd...

Podpora - 2. zgodba:

- ▶ Connected launchpad
 - ▶ nov izdelek – ne deluje stabilno ?
 - ▶ v enem letu ni rešitve



Connected LaunchPad Quick-start IoT Application dies after a day or two

We have experienced this issue on all Connected LaunchPads we have using the default application shipped on the Connected LaunchPad (CLP) and also after compiling the 'qs_iot' project and programming it to the CLP.

Essentially the device stops working after a day. I do not have a specific amount of time although it should not be hard to let run a few times to see if it is always the same.

Jul 30, 2014 7:26 PM

Mike Aanenson

Community Member

Jan 7, 2015 7:19 PM

In reply to Dubnet:

Any progress on this?

Reply

Wed, Jan 28 2015 4:35 PM

In reply to Dubnet:

Hello All,

The updated code ("qs_iot" application and underlying layers) has been under test for close to two weeks now and working. The Connected Launchpad still loses connection to Exosite Server once in a while, but successfully connects back. The system state (like on time, led state and button press state) is not lost during the connect-disconnect-connect transition. The root cause (of why connection to the Exosite Server is lost in the first place) requires analysis of network traffic when the failure occurs which is intermittent..



Stellaris Sai



Intellectual 2355 points

TI Employee

Fri, Aug 14 2015 10:56 AM



Harold Broberg



Prodigy 110 points



Community Member

I tried that one already, but am trying it again now.

It stays on line (6 minutes now). It counts button presses and shows the temperature. But when I click an LED to on, on Exosite, after a bit (23 seconds & 34 seconds, measured) the onscreen button goes back to the off position. Neither LED ever turns on, with either button.

2. Programiranje vgrajenih sistemov

Splošno :

▶ Orodja :

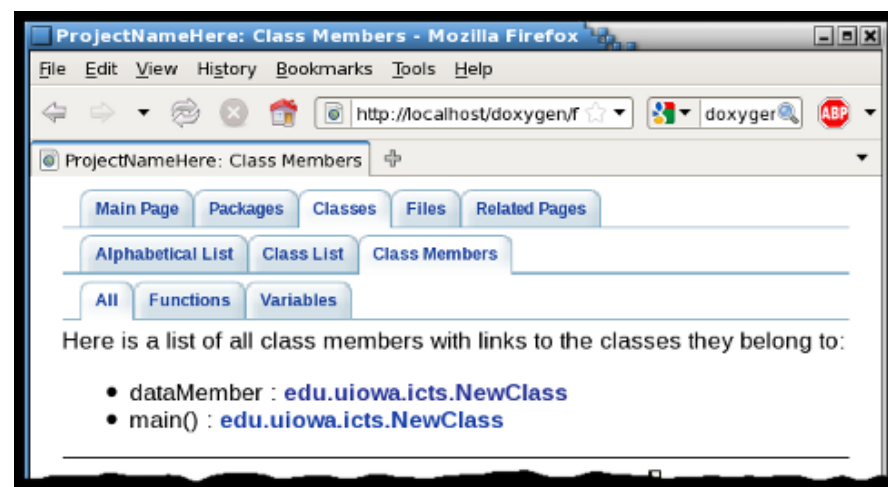
▶ IDE: CubeIDE, IAR, Keil, Eclipse

▶ Pomembne funkcionalnosti : Debug, Profile, ...

▶ Doxygen.org :

```
/**
 * @brief Short member data description.
 */
int dataMember;
```

[Primer: D13.chm](#)



Member Data Documentation

int edu.uiowa.icts.NewClass.dataMember [package]

Short member data description.

Definition at line **25** of file **NewClass.java**.

2. Programiranje vgrajenih sistemov

Splošne metode :

- ▶ Pravila robustnega programiranja
 - ▶ MISRA C ->
- ▶ Sledenje delovanja programa
 - ▶ Debugger
 - ▶ Serijska konzola
 - ▶ Log datoteka (lokalno)
- ▶ Oddaljen nadzor/logiranje (splet)

razvoj

spremljanje

```
2015-01-11 05:45:37 CRIT 232 0 FP WDT has expired
2015-01-11 05:46:21 CRIT 232 0 MNG WDT has expired
```

```
2015-01-09 15:00:02 INFO 60 0 CMDEXECUTE CMD:Execute Cmd[72]
2015-01-09 15:00:02 INFO 60 0 CMDEXECUTE CMD:SendSett
2015-01-09 15:04:02 CRIT 232 0 CMDEXECUTE WDT has expired
```



2. Programiranje vgrajenih sistemov

Pravila robustnega programiranja (preventiva)

▶ MISRA C (1998, 2004, 2012):

- ▶ **MISRA** = **M**otor **I**ndustry **S**oftware **R**eliability **A**ssociation
- ▶ 143 pravil (preverljivih z analizo) in 16 smernic
- ▶ skupine pravil:
 - ▶ **razlike** med prevajalniki (npr. velikost tipa Integer)
 - ▶ **brez funkcij s pogostejšimi napakami** (npr. malloc)
 - ▶ **obvladljiva koda** (pravila imenovanja, komentiranja...)
 - ▶ **primeri dobre prakse**
 - ▶ **omejitve kompleksnosti**
- ▶ že integrirano v nekatera IDE orodja:
 - ▶ IAR, Green Hills, ...

Rule 14.8 (required):

The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement.

The statement that forms the body of a switch statement or a while, do ... while or for loop, shall be a compound statement (enclosed within braces), even if that compound statement contains a single statement.

For example:

```
for (i = 0; i < N_ELEMENTS; ++i)
{
    buffer[i] = 0;
}
```

/* Even a single statement must be in braces */

Rule 14.4 (required):

The goto statement shall not be used.

Rule 14.5 (required):

The continue statement shall not be used.

Rule 14.6 (required):

For any iteration statement there shall be at most one break statement used for loop termination.

2. Programiranje vgrajenih sistemov - Primeri

A. Cubesensors („pametne kocke“):



Benefits Sleep Features Design

Make your home healthier,
your office more productive

Uncover the simple solutions. Just place a small, stylish, cordless
and connected Cube in each room.

Cubes are SOLD OUT!



TEMPERATURE

Your bedroom should be cooler than other rooms in your home. No more waking up covered in sweat.



HUMIDITY

Find the right balance and say goodbye to irritated throats, viruses and mold.



AIR QUALITY

Go beyond CO2. Sleep better by opening the windows and clean the air of VOCs before you go to bed.



NOISE

Light noises don't necessarily wake you up, but they can prevent you from going into deep sleep.



PRESSURE

Weather changes can also affect your body and make you restless during the night.



LIGHT

Darkness means sleep for your body. A full moon or street lights can be bright enough to start sending mix messages.



SHAKE

Shake the Cube to check your bedroom's health any time during the day.



PULSE

The Cubes give out a gentle reminder when you should start preparing for bed.



Understand your sleep data better.

You shouldn't become a sleep scientist to get a good night's sleep. With CubeSensors, you get daily practical advice that is easy to follow. Exactly when it's needed.

The CubeSensors web app is optimized for your favorite smartphone, looks stunning on your tablet or any modern web browser.

CubeSensors understand sleep tracking data from your existing sleep tracker (not included). Now available with support for Fitbit®, UP and UP24 by Jawbone™.

2. Programiranje vgrajenih sistemov - Principi

A. Cubesensors („pametne kocke“):

- ▶ Osnovni model CPU – ARM Cortex M0
- ▶ Brežžična komunikacija („Zigbee“)
- ▶ Zahteve :
 - ▶ nizka poraba, cena
- ▶ Odločitev:
 - ▶ prehod iz M3 prototipa na M0
- ▶ Posledice:
 - ▶ omejeno debugiranje
 - ▶ ni serijske konzole
 - ▶ zelo omejeni viri



2. Programiranje vgrajenih sistemov

A. Cubesensors („pametne kocke“):

Enotna **glavna zanka** (kompleksnejša izvedba) = **končni avtomat**

- ▶ **brežžična komunikacija + branje senzorjev + spanje**
- ▶ boljša organizacija kode, lažje vzdrževanje

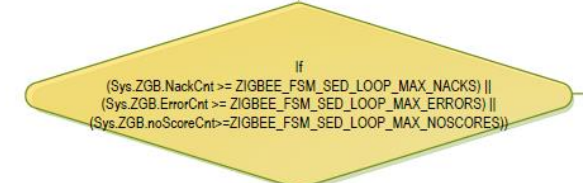
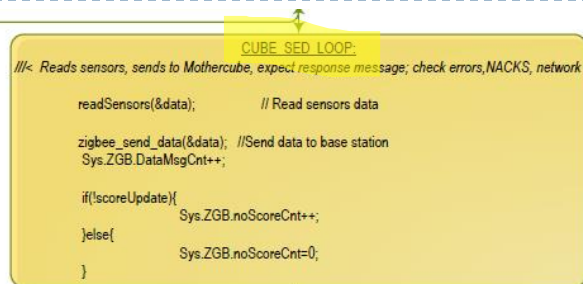


```
switch (FSM.State) {  
  
    case CHECK_POWER_ON_REASON:  
        ///  
        FSM.State: after reset or power up. SW Reset and check if it can join right away...  
  
        if VSE_OK then FSM.State = CHECK_BAUDRATE ...  
  
        break;  
  
    case CHECK_BAUDRATE:  
        ///  
        FSM.State: after reset or power up. SW Reset and check if it can join right away...  
        ...  
  
        break;  
}
```

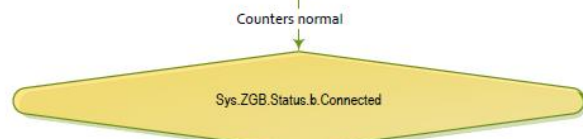
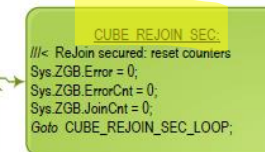
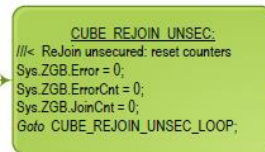
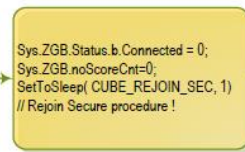
2. Programiranje vgrajenih sistemov

Cubesensors („pametne kocke“):
Diagram poteka (končni avtomat):

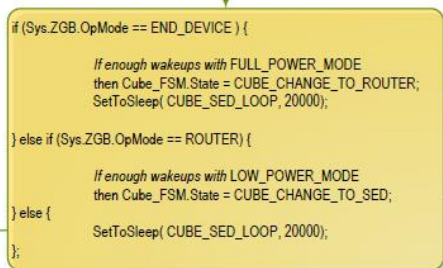
► Primer glavnega stanja



Counters High : Rejoin

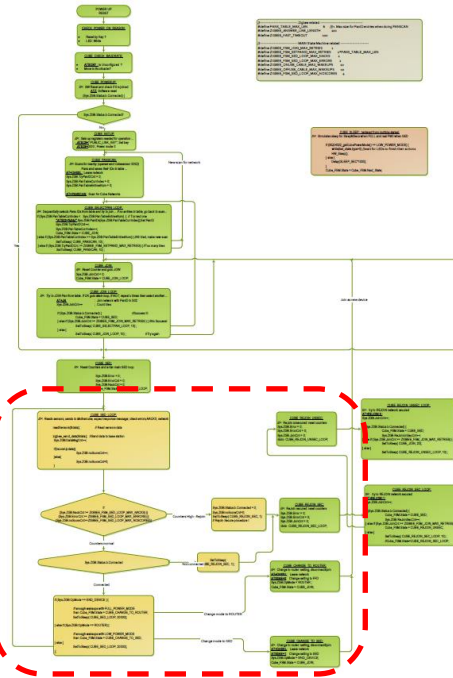
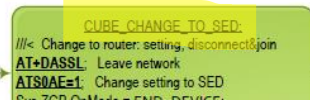
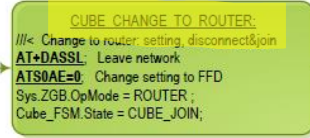


Connected



Change mode to ROUTER

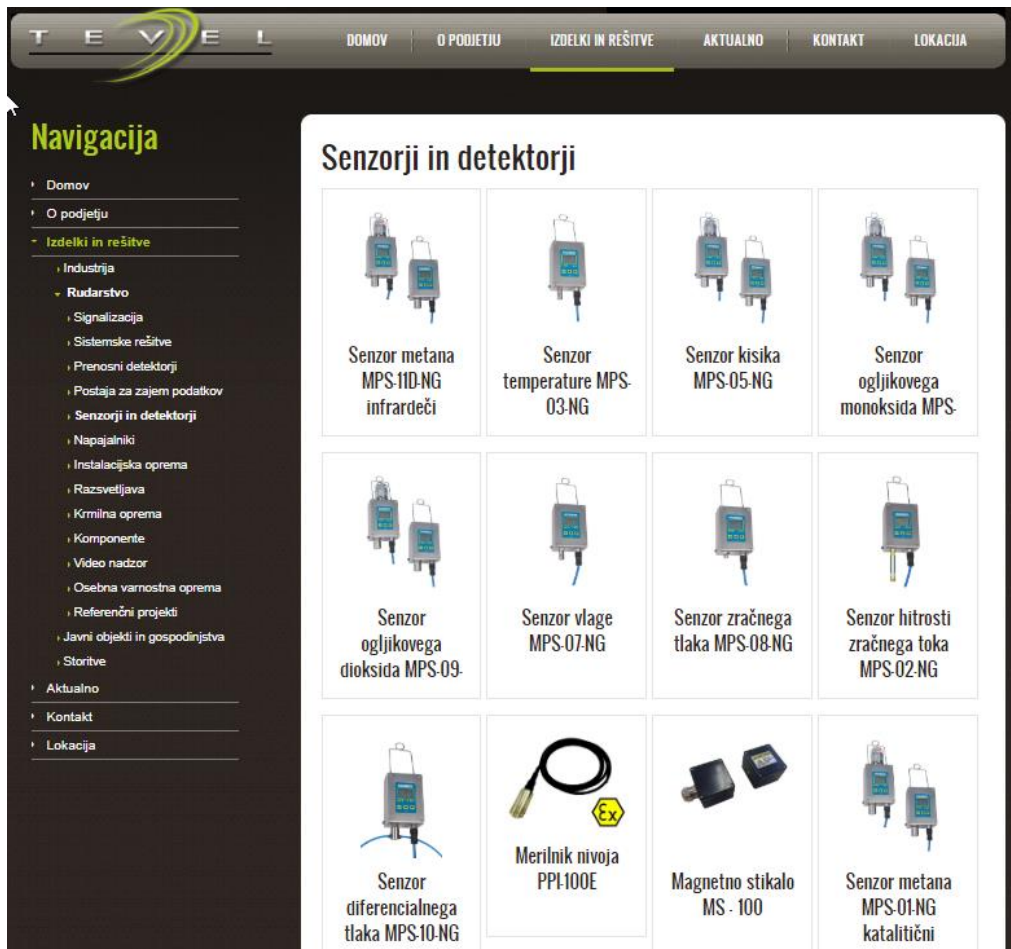
Change mode to SED



[Primer: Cube State Chart v2.pdf](#)

2. Programiranje vgrajenih sistemov

B. Tevel Pametni merilniki (rudarstvo)



The screenshot shows the Tevel website with a navigation menu on the left and a grid of sensors on the right. The navigation menu includes: Domov, O podjetju, Izdelki in rešitve (highlighted), Aktualno, Kontakt, and Lokacija. Under 'Izdelki in rešitve', there are sub-menus for Industrija, Rudarstvo (highlighted), and other categories. The 'Rudarstvo' sub-menu lists: Signalizacija, Sistemske rešitve, Prenosni detektorji, Postaja za zajem podatkov, Senzorji in detektorji (highlighted), Napajalniki, Instalacijska oprema, Razsvetljava, Krmilna oprema, Komponente, Video nadzor, Osebna varnostna oprema, Referenčni projekti, Javni objekti in gospodinjstva, and Storitve. The grid of sensors includes: Senzor metana MPS-11D-NG infrardeči, Senzor temperature MPS-03-NG, Senzor kisika MPS-05-NG, Senzor ogljikovega monoksida MPS, Senzor ogljikovega dioksida MPS-09, Senzor vlage MPS-07-NG, Senzor zračnega tlaka MPS-08-NG, Senzor hitrosti zračnega toka MPS-02-NG, Senzor diferencialnega tlaka MPS-10-NG, Merilnik nivoja PPI-100E, Magnetno stikalo MS-100, and Senzor metana MPS-01-NG katalitični.



Tevel v Kazahstanu cilja na rudarsko panogo

(video, foto) Zasavci osvajajo azijske in balkanske rudnike

Gospodarstvo Prva stran

Tevel na Kosovu

10. 10. 2018

Gospodarstvo

Tevel načrtuje nakup nemškega Wölkeja

2. Programiranje vgrajenih sistemov - Primeri

B. Tevel Pametni merilniki (rudarstvo):

Enotna **glavna zanka** – enostavnejša izvedba

```
{ ...  
  
    if (Timer_1sec) {  
        readSensors(&data); // Read sensors  
        send_data(&data);   // Send data to gateway  
        Timer_1sec = 0;  
    }  
  
    if (Timer_50msec) {  
        readKeys(&keys);    // Read user keys  
        readInputs(&inputs); // Read digital inputs  
        Timer_50msec = 0;  
    }  
  
}
```



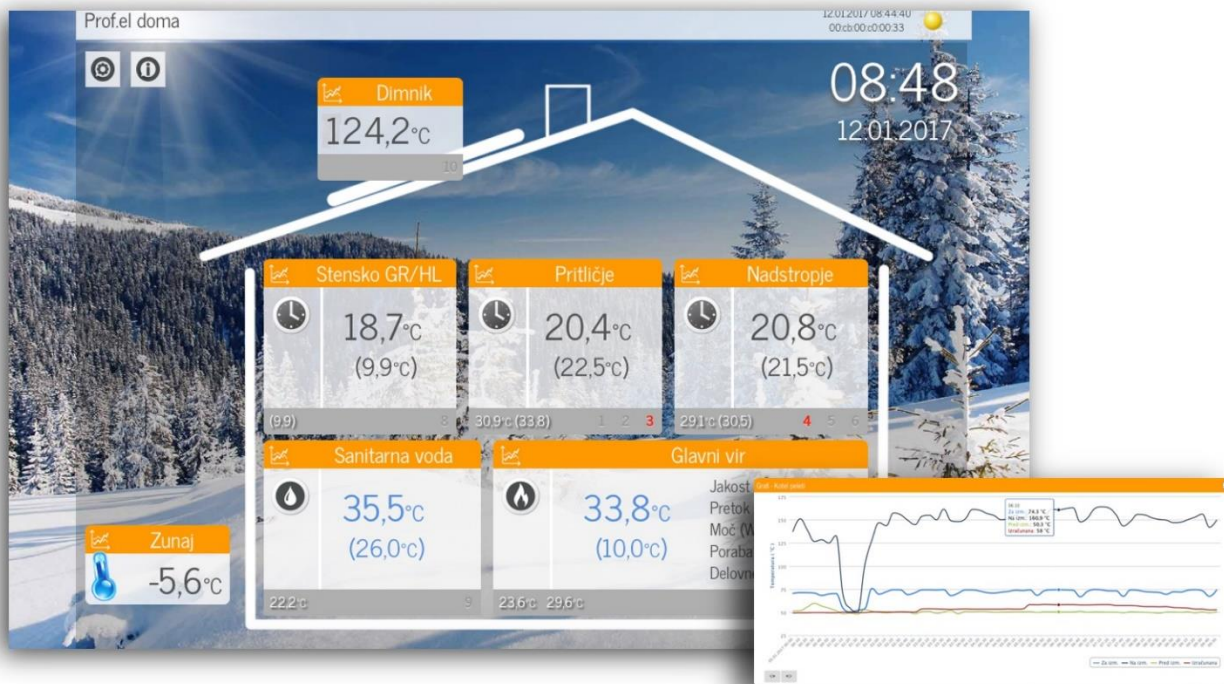
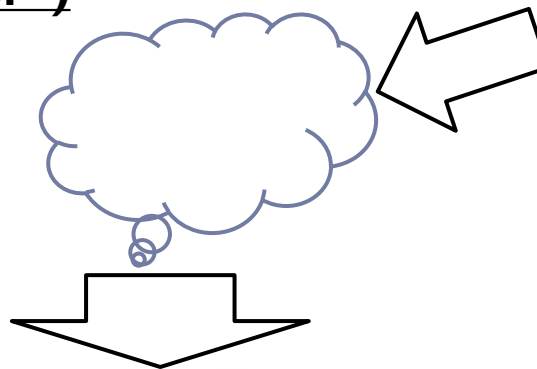
2. Programiranje vgrajenih sistemov - Primeri

C. DIALOG EQ („pametni regulator“)

RTOS – ločeni procesi

(REG,TCP,WEB,MODBUS,CANBUS)

– zahtevnejša izvedba

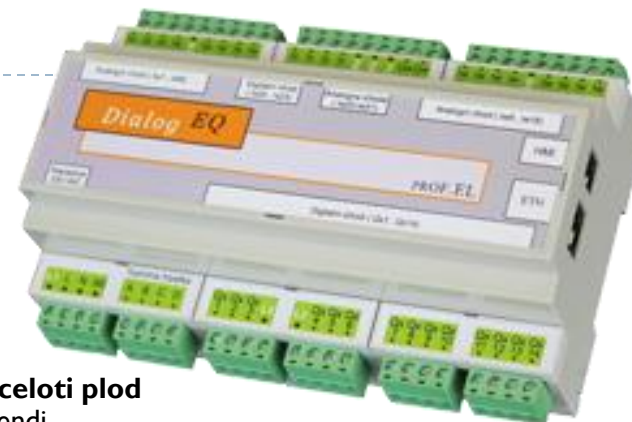


2. Programiranje vgrajenih sistemov - Primeri

C. DIALOG EQ („pametni regulator“)

RTOS – ločeni procesi (REG,TCP,WEB,MODBUS,CANBUS)

– zahtevnejša izvedba



Regulator DIALOG EQ (krajše: DEQ) predstavlja najnovejšo generacijo pametnih regulatorjev. **Je v celoti plod slovenskega znanja.** V podjetju PROF.EL smo ga razvili, saj želje strank in pa predvsem sodobni trendi narekujejo daljinski nadzor in avtomatizacijo ogrevalno/hladilnih sistemov oziroma celovite rešitve za pametni dom (smart house).

Regulator DEQ vam omogoča:

- ▶ 24h **spremljanje delovanja in upravljanje s sistemom na daljavo** (računalnik, tablica, pametni telefon),
- ▶ varno **shranjevanje** vseh uporabnikovih nastavitev v oblaku,
- ▶ pregled in analizo delovanja sistema,
- ▶ prijazen **uporabniški vmesnik** (interni WEB, WEB aplikacija, aplikacija za mobilne telefone Android, iOS, Windows),
- ▶ beleženje in shranjevanje podatkov (črna skrinjica v oblaku),
- ▶ **daljinsko pomoč** servisne ekipe,
- ▶ **daljinsko posodabljanje** (up-grade) programske opreme.

Algoritmi so pripravljeni za vodenje in nadzor kotlov na biomaso in olje, toplotnih črpalk, sončnih kolektorjev, sanitarne vode s cirkulacijsko črpalko idr.

2. Programiranje vgrajenih sistemov - Primeri

C. DIALOG EQ („pametni regulator“)

D13 („HVAC regulator“)

Izhodi in algoritmi za krmiljenje:

- ▶ Direktne veje
- ▶ Mešalne veje 2x
- ▶ Sanitarne vode
- ▶ Sončnih kolektorjev

Kompleksnejša izvedba:

- ▶ **MQX RTOS**
- ▶ **Opravila :**
 - ▶ FP_TASK glavni krmilni program
 - ▶ MODBUS_TASK Modbus strežnik
 - ▶ TCPCLIENT_TASK povezava s podatkovnim strežnikom v oblaku
 - ▶ httpd_server spletni strežnik – lokalni portal
 - ▶ CMDEXECUTE_TASK izvedba ukazov
 - ▶ FTPCLIENT_TASK FTP prenosi



2. Programiranje vgrajenih sistemov - Primeri

C. DIALOG EQ („pametni regulator“)

RTOS (primer MQX) :

Opravila („Tasks“)

```
const TASK_TEMPLATE_STRUCT MQX_template_list[] =
{
    /* Task Index,   Function,      Stack, Priority,           Name,                Attributes,           Param, Time Slice */
    { MNG_TASK,      MngTask,        1200, TASK_PRIORITY_MNG_TASK, MNG_TASK_DES,        MQX_AUTO_START_TASK, 0,      0 },

    { SHELL_TASK,    ShellTask,      2000, TASK_PRIORITY_SHELL,    SHELL_TASK_DES,      0,      0 },

    { FP_TASK,       FunPgmTask,     2000, TASK_PRIORITY_FP,       FP_TASK_DES,         0,      0 },

    { TNSH_TASK,     TelnetsClientShell, 2000, TASK_PRIORITY_TNETSH, TNSH_TASK_DES,      0,      0 },

    { TCPCLIENT_TASK, TCPClient_Task, 2000, TASK_PRIORITY_TCPCLIENT, TCPCLIENT_TASK_DES, 0,      0 },

    { MODBUS_TASK, Modbus_Task, 2000, TASK_PRIORITY_MODBUS, MODBUS_TASK_DES, 0,      0 },

    { EVTALM_TASK, EventAlmTask, 2000, TASK_PRIORITY_EVTALM, EVTALM_TASK_DES, 0,      0 },

    { AIN_TASK,      AinTask,        500,  TASK_PRIORITY_AIN,      AIN_TASK_DES,        0,      0 },

    { NETMNG_TASK, NetMngTask, 1000, TASK_PRIORITY_NETMNG, NETMNG_TASK_DES,     0,      0 },

    { 0 }
};
```

2. Programiranje vgrajenih sistemov - Primeri

C. DIALOG EQ („pametni regulator“)

RTOS (primer MQX opravila) :

Glavna regulacijska zanka („FP TASK“)

```
void FunPgmTask (uint_32 initial_data)
{
    FunPgmInit();

    // register task for system messages
    rc = SysMsgRegister ();

    // WDT control
    WdtRegister (15000, WDT_ACTION_LOG);

    // ----- main execution loop -----
    while (TRUE) {

        _time_get_elapsed (&fp_start_time); //Measure processing time fp_start_time

        WdtReset ();

        FunPrepareFPData();    // Prepare FP data
        FunRegulation();       // Iterate regulation loops
        FunCommitFPData();     // Commit any changes back to system

        _time_get_elapsed (&fp_end_time); //Measure processing time
        _time_diff (&fp_start_time, &fp_end_time, &fp_loop_time); // get elapsed time
        FPLoopTime=(fp_loop_time.SECONDS * 1000) + fp_loop_time.MILLISECONDS;

        _time_delay(1000-FPLoopTime);           // wait for 1000 ms - loop time in ms
    }

    _task_block();           // Shouldn't reach this point
}
```

```
/** @brief FP: Main Functional Program Task.
    Calls FunPgmInit for initialization and then runs endless main FP loop.
    *
    * This is main functional program task.
    * It will first run Initializations: FunPgmInit();
    * Then it will proceed in endless loop :
    *     FunPrepareFPData(); // Prepare FP data
    *     FunRegulation();    // Iterate regulation loops
    *     FunCommitFPData();  // Commit any changes back to system
    *                         check if settings changed - if yes, then read all settings
    */
```

```
void FunPgmTask ( uint_32 initial_data )
```

FP: Main Functional Program Task. Calls FunPgmInit for initialization and then runs endless main FP loop.

This is main functional program task. It will first run Initializations: **FunPgmInit()**; Then it will proceed in endless loop : **FunPrepareFPData()**; // Prepare FP data **FunRegulation()**; // Iterate regulation loops **FunCommitFPData()**; // Commit any changes back to system check if settings changed - if yes, then read all settings

Todo:

Temporary - shouldn't be used in production code !!!

Definition at line 139 of file **fp.c**.

References **APPCFG_DEFAULT_FP_USER_ACCCODE**, **APPCFG_PRINTF**, **D13_GVARS::Day**, **FunCommitFPData()**, **FunLogCurrentState()**, **FunPgmInit()**, **FunPrepareFPData()**, **FunRegulation()**, **FunSimCommitFPData()**, **FunSimLogCurrentState()**, **FunSimPgmInit()**, **FunSimPrepareFPData()**, **FunSimRegulation()**, **FP_DATA::GVars**, **D13_GVARS::Hour**, **D13_GVARS::Minute**, **D13_GVARS::Month**, **Read_FPSettings()**, **D13_GVARS::Second**, and **D13_GVARS::Year**.



D. Tinia – prijazen dom

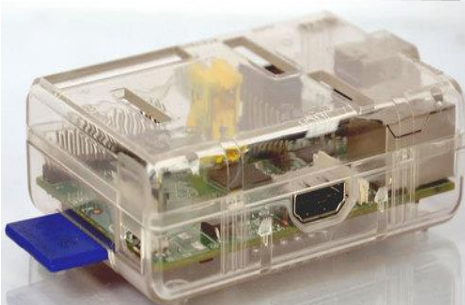
TBS – „Tinia Building Server”

Kratek opis

TBS – „Tinia Building Server”:

Nadzor, upravljanje in vizualizacija delovanja prijaznega doma.

- majhen, varčen, tih (5W)
- povezuje zgradbo in pametno mesto
- informiranje, povratna inf.
 - pametni telefoni, tablice
 - splet, soc.omrežja
- programiranje s pravili, vtičniki
- povezava s soc.omrežji
 - Twitter, FaceBook

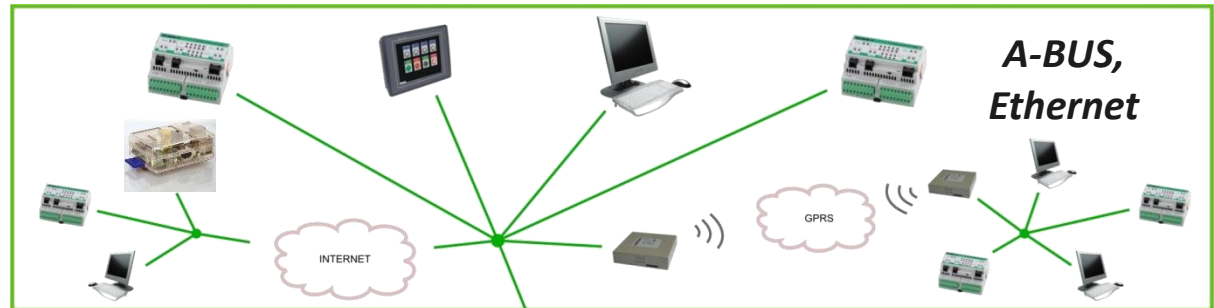


INTEGRA BM SYSTEM

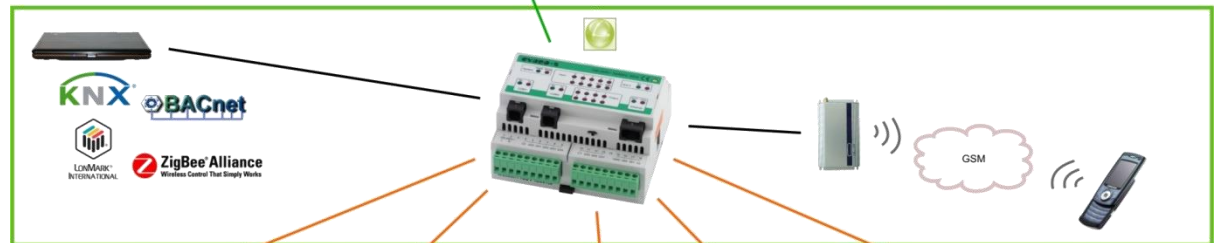
Industrial & Building Automation

Generally

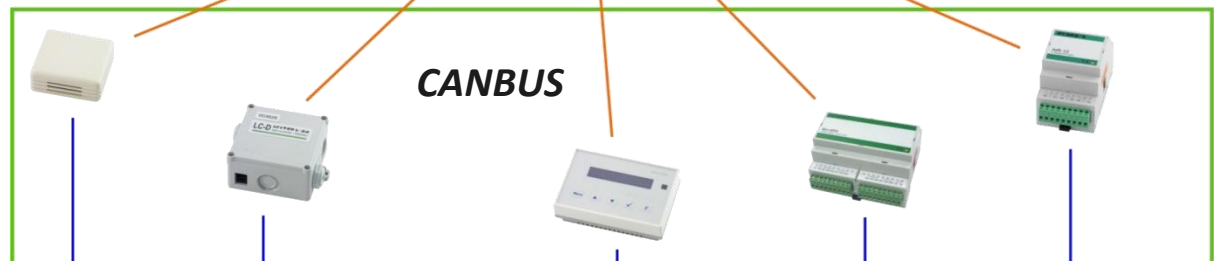
High level network



CyBro controller



Low level network



Accessories





Tinia – prijazen dom

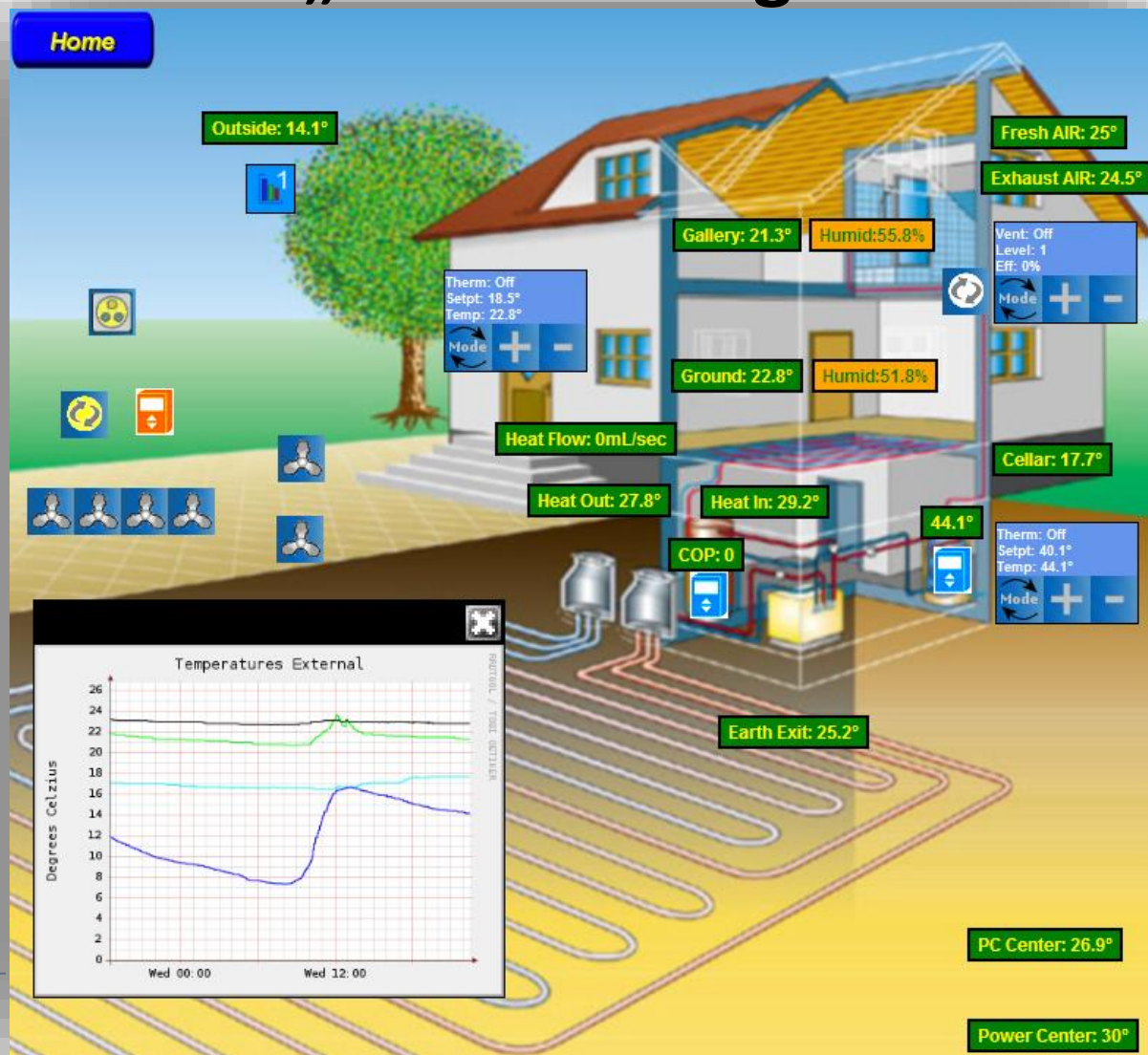
TBS – „Tinia Building Server”

Kratek opis

TBS – „Tinia Building Server”:

Nadzor, upravljanje in vizualizacija delovanja prijaznega doma.

- majhen, varčen, tih (5W)
- povezuje zgradbo in pametno mesto
- informiranje, povratna inf.
 - pametni telefoni, tablice
 - splet, soc.omrežja
- programiranje s pravili, vtičniki
- povezava s soc.omrežji
 - Twitter, FaceBook





Ogrevanje (prostori, sanitarna voda)

Toplotna črpalka zemlja-voda

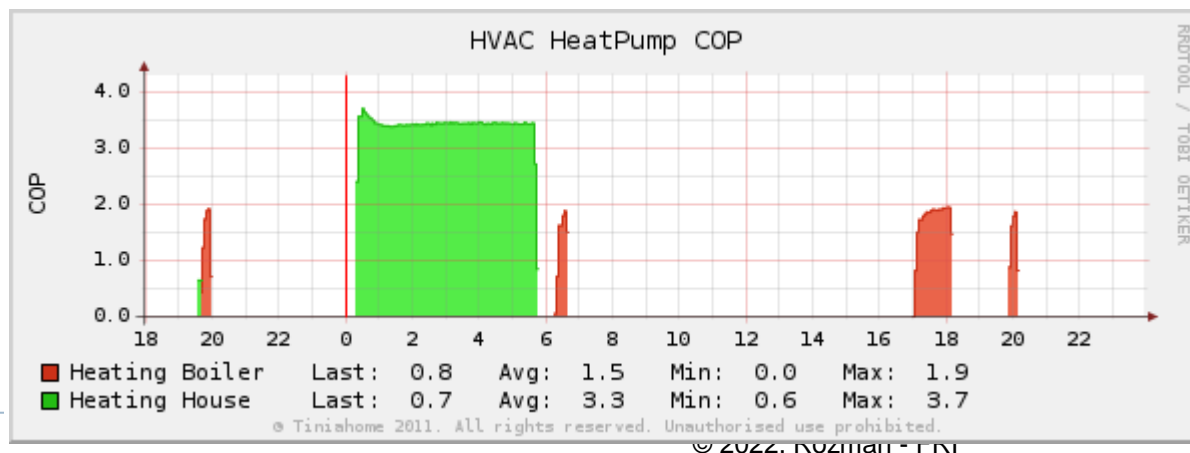
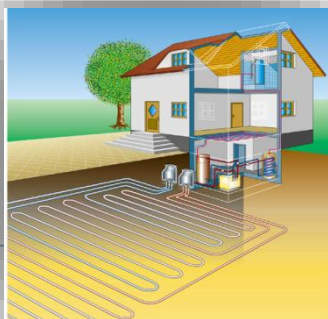
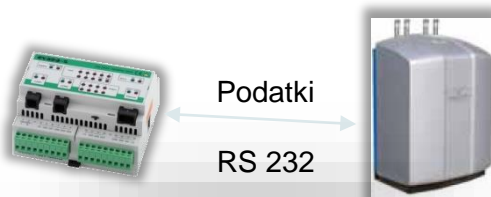
- Zemeljski kolektor
- Talno in stensko ogrevanje
- Sanitarna voda
- Serijska komunikacija:
Cybro COM2 <-> TČ

Grelno število (COP-Coefficient Of Performance):

$$COP = \frac{\text{Toplotna Moč}}{\text{Elektricna Moč}}$$

- COP ~ 3.5 Ogrevanje prostorov (Elekt.Moč =1.8 kW)
- COP ~ 2.0 Ogrevanje sanitarne vode (Elekt.Moč =3.0 kW)

Primer zimskega dneva – COP toplotne črpalke :





Prezračevanje

Prezračevanje s povratkom toplote (rekuperacija)

- Izčrpan zrak odda toploto svežemu zraku s >85% izkoristkom
- Vkllop/izklop in kontrola pretoka
- Indikator zasičenosti filtra

Učinkovitost rekuperacije:

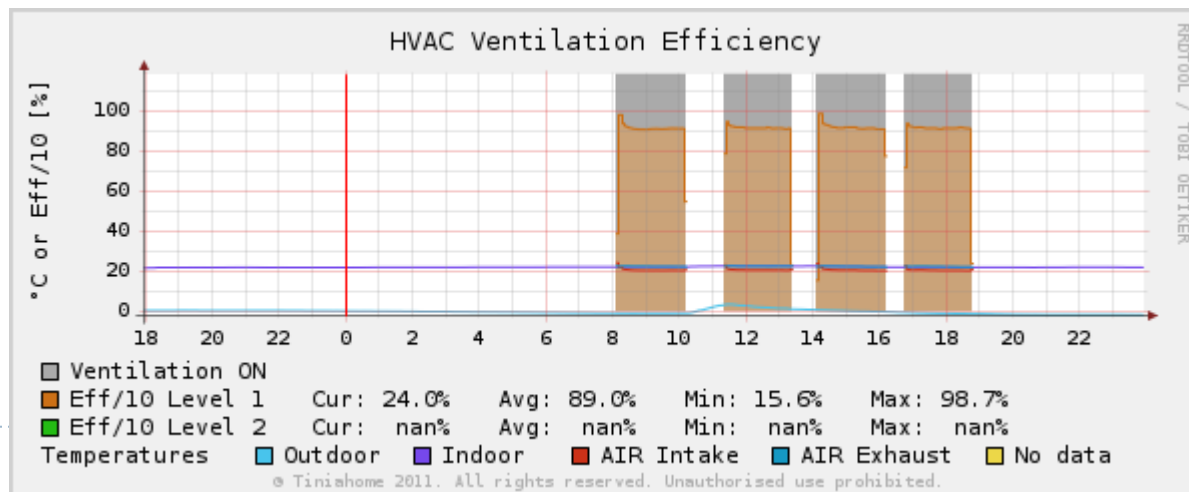
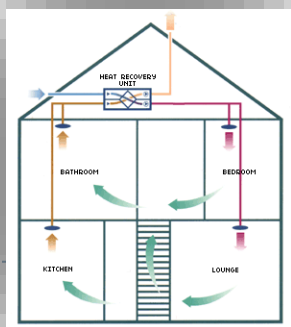
$$EFF \approx \frac{\text{Svež zrak temp.} - \text{Zunanji zrak temp.}}{\text{Izčrpan zrak temp.} - \text{Zunanji zrak temp.}} [\%]$$



Primer : Eff ~ 90% ko:

- Zunanja Temp. ~0°C
- Notranja Temp. ~21°C
- Sveži zrak se segreje od ~0°C do ~19°C (rekuperacija)

Primer zimskega dneva – učinkovitost rekuperacije





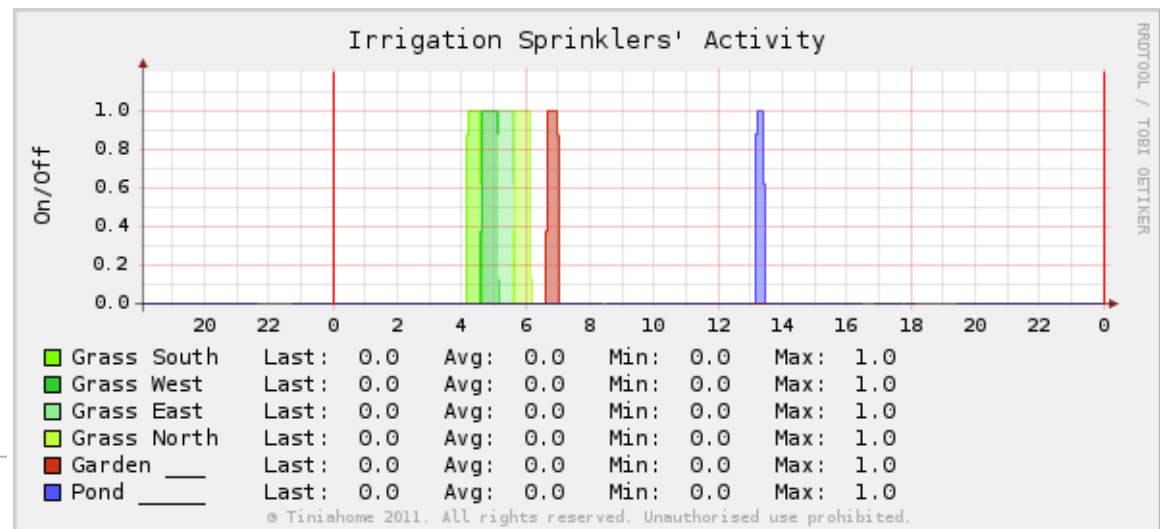
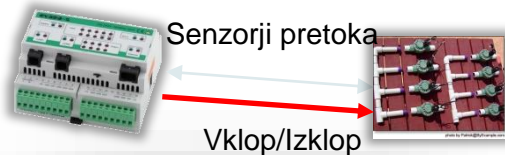
Zalivanje (vrt, zelenica, ribnik)

Zalivanje (vrt, zelenica, ribnik)

- Kontrola zalivanja v skladu z **urnikom** in **nivojem vlage** v tleh
- Vkllop/Izklop & Zaznavanje pretoka

- Zaznavanje pretoka:
 - Preklop med dvema viroma :
 - podtalnica
 - vodovod

Primer poletnega dne - Zalivanje





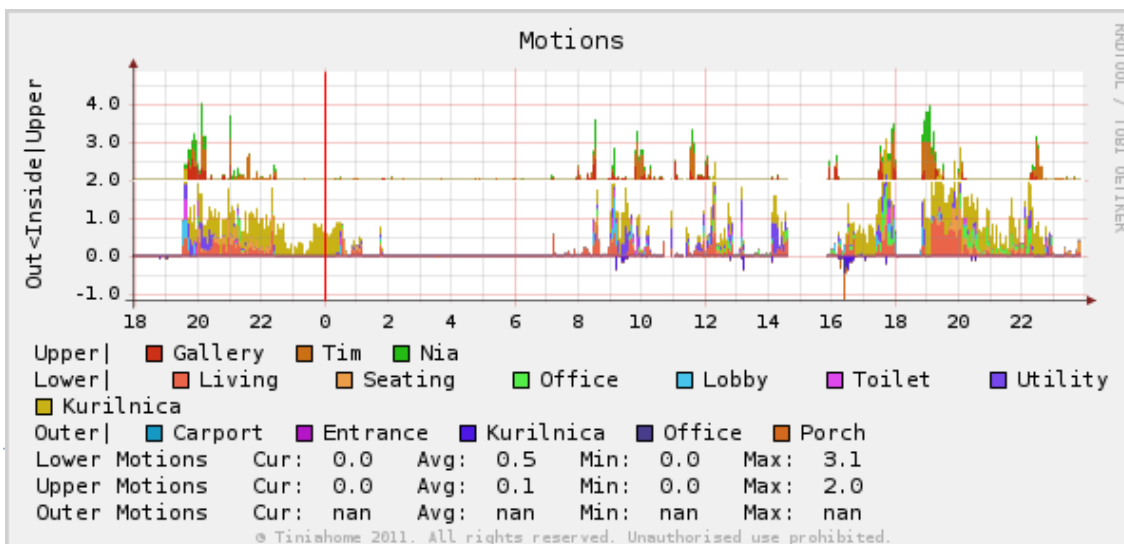
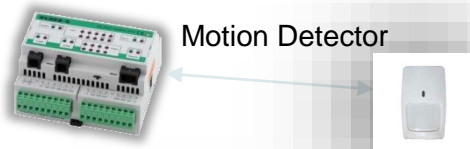
Vzorci gibanja - prisotnost

Detekcija gibanja/prisotnosti

- Detekcija gibanja v posameznih prostorih
- Informacija o prisotnosti

Uporaba vzorcev gibanja oziroma prisotnosti :

- Upravljanje :
 - Razsvetljave
 - Ogrevanja, hlajenja
 - A/V naprav
- Profiliranje, napovedi :
 - Dogodkov v prihodnosti
 - Energetskih potreb
 - Nastavitev





Razsvetljava

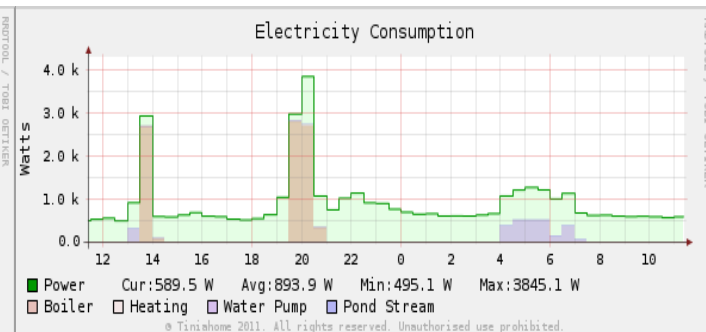
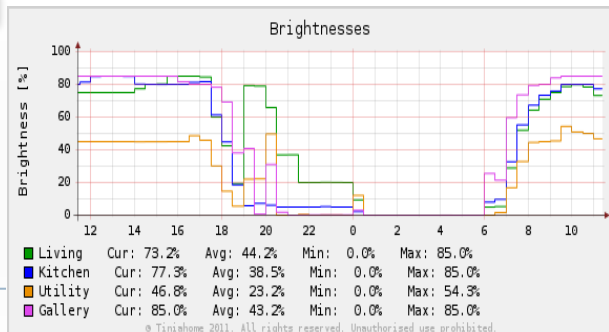
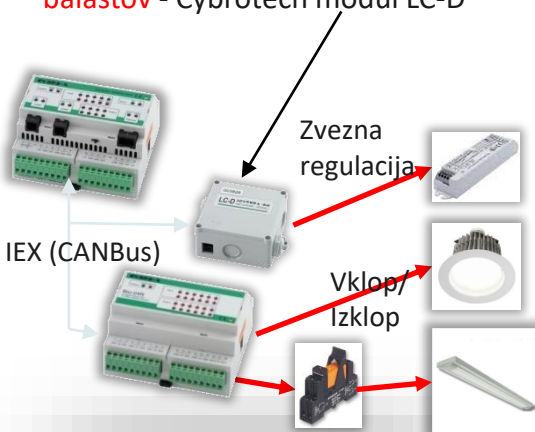
Vklop/Izklop in zvezna regulacija razsvetljave

• **Vklop/Izklop** kontrola s
Cybrotech moduli BIO-24R in BIO-24T, Zunanji releji

• **Zvezna regulacija** s pomočjo **DALI balastov** - Cybrotech modul LC-D

- Luči se upravljajo v skupinah
- Običajno krmiljena s pomočjo scen in zaznavanje osvetljenosti:
 - **Statične scene** – npr. : Prehrana, Obisk, Romantika, TV, Branje, Relaksacija, ...
 - **Dogodkovne scene**: Ko se vklopi TV, nastavi bližnjo luč na 20%.
- Zmanjševanje porabe :
 - **Časovne luči** (izklopi po določenem času odsotnosti)
 - Vklopi luč samo, ko je to **res potrebno** (trenutna osvetljenost)
 - Nastavi zvezne luči samo na **potrebno stopnjo** (glede na osvetljenost)

Primer **meritev osvetljenosti in nadzora porabe el. energije**
(glavni porabnik el. Energije so posebej izpostavljeni)



Pasivno ogrevanje/hlajenje...

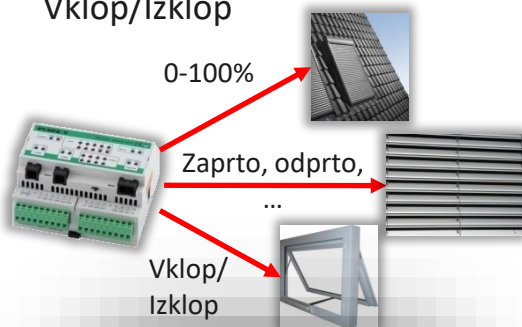


Rolete, žaluzije, Okna

• **Rolete:** med 0% - 100%
(0% odprte, 100% zaprte)

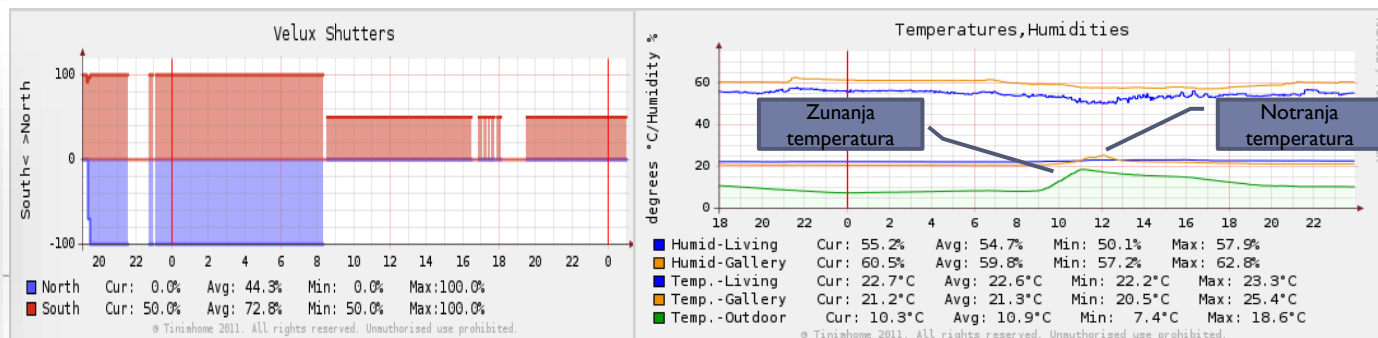
• **Žaluzije** imajo stanja :
Zaprto(100%), Senčeno(75%),
Odprto(50%), Solarno pasivno
(25%), Dvignjeno(0%).

• **Motorizirana okna:**
Vkllop/Izklop



- **Strešna okna z roletami :**
 - Severna, običajno:
 - **Odprta v toplem vremenu** za boljšo osvetlitev
 - **Zaprta v hladnem vremenu** za ohranjanje toplote
 - Južna, običajno:
 - **Odprta v hladnem, sončnem vremenu** za pasivno ogrevanje
 - **Zaprta v vročem vremenu** proti pregrevanju
- **Žaluzije:**
 - **Senčene ali zaprte ob izrazitem sončnem vremenu poleti**
 - **Odprte v "solarni" poziciji ob sončnih dnevih pozimi**
- **Motorizirana okna (s komarniki) :**
 - **Odprta v poletnih nočeh za pasivno ohlajanje**

Primer stanj rolet in temperatur v sončnem zimskem dnevu:



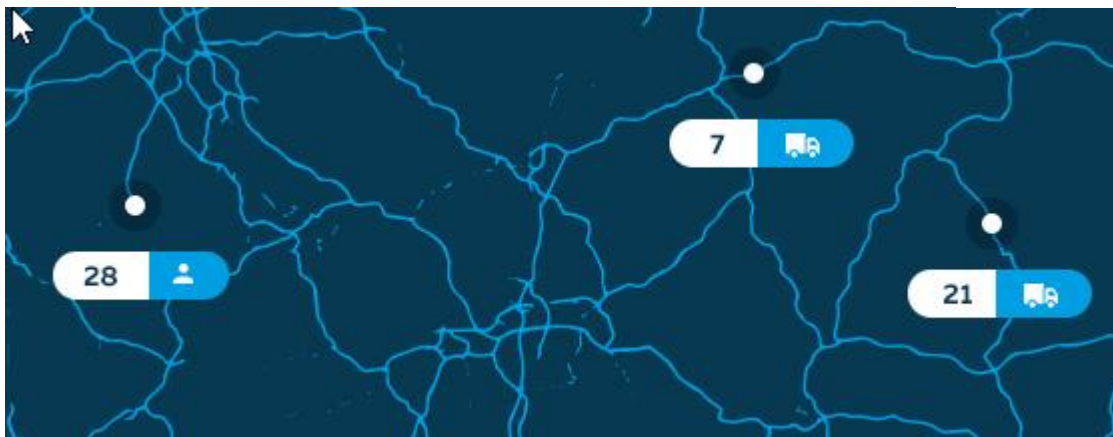
2. Programiranje vgrajenih sistemov - Primeri

E. Pametni zabojnik

Merilnik tlaka, temperature, nivoja, pozicije, ... za pametne zabojnike



Füllstand	Raumtemperatur	Raumfeuchtigkeit	Behälterdruck
Licht	Ortung	Beschleunigung	Außentemperatur
Umgebungsfeuchtigkeit	Luftdruck	Umgebungslicht	



2. Programiranje vgrajenih sistemov - Primeri

E. Pametni zabojujnik

Merilnik tlaka, temperature, nivoja, pozicije, ... za pametne zabojujnik

STM32CubeMX CubeMX_F439_Devkit_FreeRTOS.ioc: STM32F439ZITx

File Window Help

Home > STM32F439ZITx > CubeMX_F439_Devkit_FreeRTOS.ioc - Pinout & Configuration > GENERATE CODE

Pinout & Configuration Clock Configuration Project Manager Tools

Additional Software Pinout

Categories A-Z

System Core

- DMA
- GPIO
- IWDG
- NVIC
- ▲ RCC
- ▼ SYS
- WWDG

Analog >

Timers >

Connectivity >

Multimedia >

Security >

- CRYP
- HASH
- ✓ RNG

Computing >

Middleware >

- ✓ FATFS
- ✓ FREERTOS
- GRAPHICS
- LIBJPEG
- ✓ LWIP
- ✓ MBEDTLS
- PDM2PCM
- ✓ USB_DEVICE
- USB_HOST

FREERTOS Mode and Configuration

Interface CMSIS_V1

Configuration

Reset Configuration

- ✓ Mutexes
- ✓ FreeRTOS Heap Usage
- ✓ MPU Settings
- ✓ Tasks and Queues
- ✓ Timers and Semaphores
- ✓ Config parameters
- ✓ Include parameters
- ✓ User Constants

Configure the following parameters:

Search (Ctrl+F)

FreeRTOS API CMSIS v1

Versions

- FreeRTOS version 10.0.1
- CMSIS-RTOS version 1.02

Kernel settings

- USE_PREEMPTION Enabled
- CPU_CLOCK_HZ SystemCoreClock
- TICK_RATE_HZ 100
- MAX_PRIORITIES 7
- MINIMAL_STACK_SIZE 128 Words
- MAX_TASK_NAME_LEN 16
- USE_16_BIT_TICKS Disabled
- IDLE_SHOULD_YIELD Enabled
- USE_MUTEXES Enabled
- USE_RECURSIVE_MUTEXES Disabled
- USE_COUNTING_SEMAPHORES Enabled
- QUEUE_REGISTRY_SIZE 8
- USE_APPLICATION_TASK_TAG Disabled
- ENABLE_BACKWARD_COMPATIBILITY Enabled
- USE_PORT_OPTIMISED_TASK_SELECTION Enabled

USE_PREEMPTION

configUSE_PREEMPTION

Parameter Description:

Set to 1 to use the preemptive RTOS scheduler, or 0 to use the cooperative RTOS scheduler.

Pinout view System view

STM32F439ZITx LQFP144

screenrec

2. Programiranje vgrajenih sistemov - Primeri

F. Embedded Linux (UcLinux, Buildroot)

Buildroot na STM32F769

The logo for Bootlin, featuring the word "bootlin" in a lowercase, sans-serif font. A small orange triangle is positioned above the letter "i".

[Understanding the Linux graphics stack](#)

[Running Linux with PM2MP1 Kit](#)

<https://bootlin.com/>



THE YOCTO PROJECT. IT'S NOT AN EMBEDDED LINUX DISTRIBUTION,
IT CREATES A CUSTOM ONE FOR YOU.



Products

NXP,
Cortex-A

- i.MX 8M Mini
- i.MX 8M
- i.MX 6SoloX
- i.MX 6ULL
- Vybrid

NXP,
i.MX 8M Mini Starter Kits

- NXP 8MNavQ Kit
- PMD TOF Camera Kit

NXP,
Cortex-M

- i.MX RT1050
- i.MX RT1060
- i.MX RT1170
- Kinetis K70
- Kinetis K61
- LPC4357
- LPC4350
- LPC1850
- LPC1788

ST,
Cortex-A

- STM32MP1

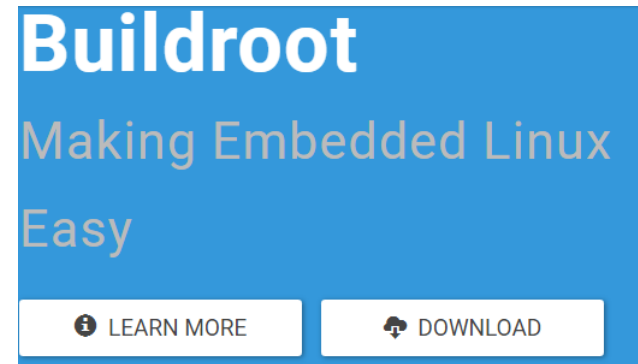
ST,
Cortex-M

- STM32H7
- STM32F7
- STM32F4
- STM32F769I
- STM32F746G
- STM32F429

Microchip,
Cortex-M

- SmartFusion2
- SmartFusion

<https://www.emcraft.com/>

A blue banner for Buildroot. The word "Buildroot" is in large, white, bold, sans-serif font. Below it, "Making Embedded Linux" is in a smaller, white, sans-serif font, and "Easy" is in a larger, white, sans-serif font. At the bottom, there are two white buttons: "LEARN MORE" with an information icon and "DOWNLOAD" with a download icon.

2. Programiranje vgrajenih sistemov - Primeri

F. Embedded Linux (UcLinux, Buildroot)

Buildroot na STM32F769

```
# -----  
U-Boot SPL 2020.04 (Oct 18 2020 - 20:10:10 +0200)  
# -----  
Trying to boot from XIP  
|  
|  
U-Boot 2020.04 (Oct 18 2020 - 20:10:10 +0200)
```

Model: STMicroelectronics STM32F769-DISCO board

DRAM: 16 MiB

set_rate not implemented for clock index 4

set_rate not implemented for clock index 4

set_rate not implemented for clock index 4

Flash: 1 MiB

MMC: sdio2@40011c00: 0

In: serial

Out: serial

Err: serial

usr button is at LOW LEVEL

Net:

Warning: ethernet@40028000 (eth0) using random MAC

eth0: ethernet@40028000

Hit SPACE in 1 seconds to stop autoboot:

```
# -----
```

Starting kernel ...

```
# -----
```

Booting Linux on physical CPU 0x0

Linux version 5.6.15 (robi@Linux) (gcc version 8.4.0 (Buildroot 2020.05)) #1 PREEM

CPU: ARMv7-M [411fc270] revision 0 (ARMv7M), cr=00000000

CPU: PIPT / VIPT nonaliasing data cache, PIPT instruction cache

OF: fdt: Machine model: STMicroelectronics STM32F769-DISCO board

Reserved memory: created DMA memory pool at 0xc0ef0000, size 1 MiB

OF: reserved mem: initialized node linux,dma-compatible id shared-dma-pool

Using ARMv7 PMSA Compliant MPU. Region independence: No, Used 6 of 8 regions

Built 1 zonelists, mobility grouping off. Total pages: 3794

Kernel command line: root=/dev/mmcblk0p1 rootwait rw

Dentry cache hash table entries: 2048 (order: 1, 8192 bytes, linear)

Inode-cache hash table entries: 1024 (order: 0, 4096 bytes, linear)

mem auto-init: stack:off, heap alloc:off, heap free:off

Memory: 12240K/15296K available (1924K kernel code, 166K rwdata, 384K rodata, 84K init, 115K bss, 3056K reserved,
0K cma-reserved)

Differences between UCLinux vs Linux

Maybe we can have session to explain what is the biggest difference...

Under MMU, every program runs in own address space, independently of others, it can also ask for more memory any time during execution...

Under UCLinux this is not the case, you have linear memory space for all apps, including Kernel... If one app goes wrong, it can affect others, since memory space is the same... No protection at all for this situation... This is the biggest difference.... And also for UCLinux, at least 32MB of RAM is recommended, at least to start with...

But I also see some real advantages of using UCLinux already...

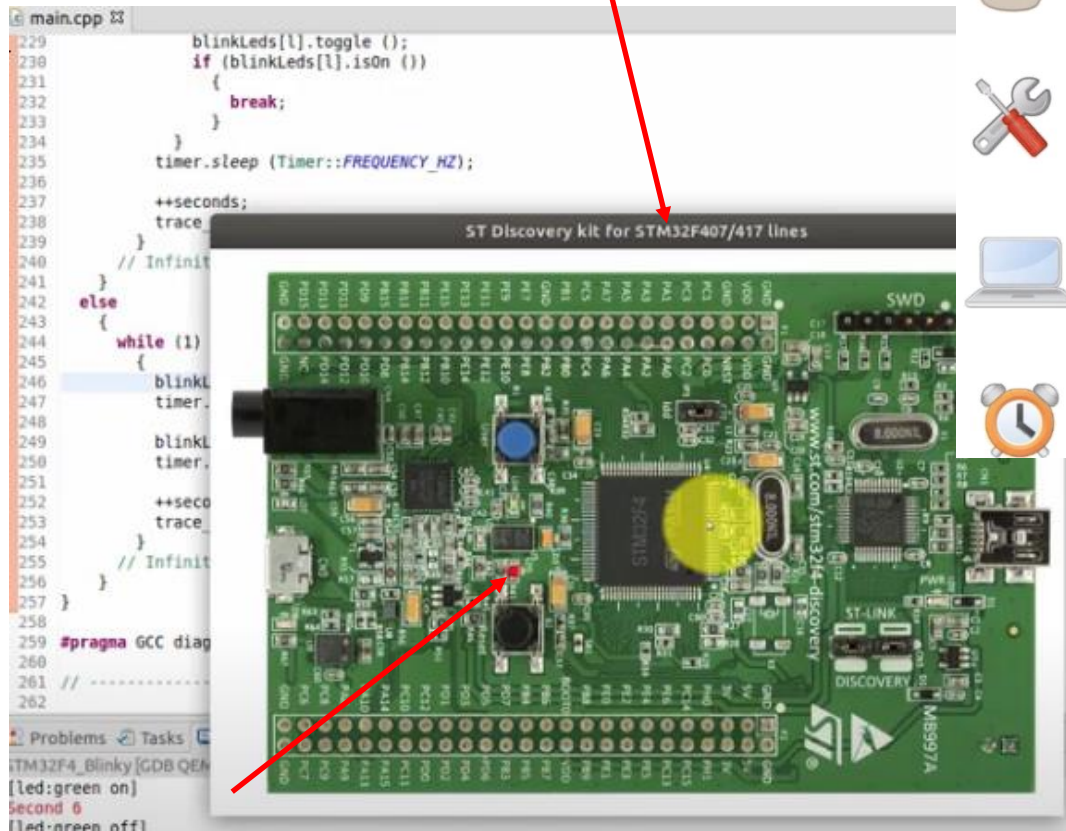
Because SW looks quite the same, we can use tools that are already existing... We can develop on Linux and transfer.... So SW very much looks like real Linux SW. And that is the big advantage of this path... So I still think that it is a good way... But we need to also have practical experience, when system will be actually running to confirm all this...

2. Programiranje vgrajenih sistemov - Primeri

G. Simulacije, Emulacije

QEMU – STM32 Discovery

<https://www.qemu.org/>



Why use QEMU?

QEMU

A generic and open source machine emulator and virtualizer



• Cost

- free and open source software (GPLv2)
- no development kit required



• Experiment without fear

- Minimize the risk of corrupting valuable development boards



• Portability

- Not tied to a lab bench --> only need QEMU and a laptop



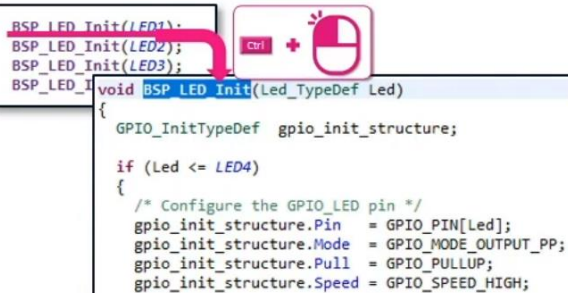
• Reduce project timescales

- work in advance of prototype board or silicon delivery

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelIDE)

Symbol Hyperlink

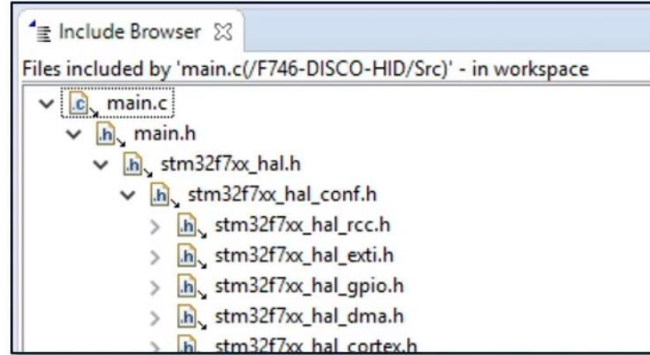


```
BSP_LED_Init(LED1);
BSP_LED_Init(LED2);
BSP_LED_Init(LED3);
BSP_LED_Init(LED4);

void BSP_LED_Init(Led_TypeDef Led)
{
    GPIO_InitTypeDef gpio_init_structure;

    if (Led <= LED4)
    {
        /* Configure the GPIO_LED pin */
        gpio_init_structure.Pin = GPIO_PIN[Led];
        gpio_init_structure.Mode = GPIO_MODE_OUTPUT_PP;
        gpio_init_structure.Pull = GPIO_PULLUP;
        gpio_init_structure.Speed = GPIO_SPEED_HIGH;
```

Include Browser

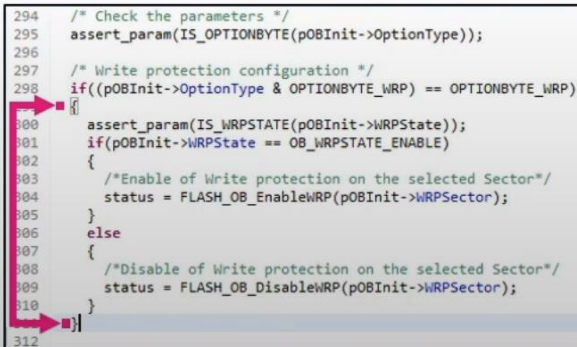


Include Browser

Files included by 'main.c (/F746-DISCO-HID/Src)' - in workspace

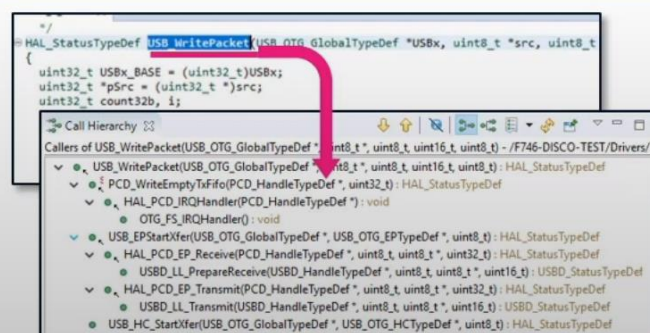
- main.c
 - main.h
 - stm32f7xx_hal.h
 - stm32f7xx_hal_conf.h
 - stm32f7xx_hal_rcc.h
 - stm32f7xx_hal_exti.h
 - stm32f7xx_hal_gpio.h
 - stm32f7xx_hal_dma.h
 - stm32f7xx_hal_cortex.h

Brace Navigation



```
294 /* Check the parameters */
295 assert_param(IS_OPTIONBYTE(pOBInit->OptionType));
296
297 /* Write protection configuration */
298 if((pOBInit->OptionType & OPTIONBYTE_WRP) == OPTIONBYTE_WRP)
299 {
300     assert_param(IS_WRPSTATE(pOBInit->WRPState));
301     if(pOBInit->WRPState == OB_WRPSTATE_ENABLE)
302     {
303         /*Enable of Write protection on the selected Sector*/
304         status = FLASH_OB_EnableWRP(pOBInit->WRPSector);
305     }
306     else
307     {
308         /*Disable of Write protection on the selected Sector*/
309         status = FLASH_OB_DisableWRP(pOBInit->WRPSector);
310     }
311 }
312
```

Call Hierarchy



```
HAL_StatusTypeDef USB_WritePacket(USB_OTG_GlobalTypeDef *USBx, uint8_t *src, uint8_t *dest)
{
    uint32_t USBx_BASE = (uint32_t)USBx;
    uint32_t *pSrc = (uint32_t *)src;
    uint32_t count32b, i;

    Callers of USB_WritePacket(USB_OTG_GlobalTypeDef *USBx, uint8_t *src, uint8_t *dest) - /F746-DISCO-TEST/Drivers/USB
    - USB_WritePacket(USB_OTG_GlobalTypeDef *USBx, uint8_t *src, uint8_t *dest)
    - PCD_WriteEmptyTxFifo(PCD_HandleTypeDef *hpcd, uint32_t *pTxFifo)
    - HAL_PCD_IRQHandler(PCD_HandleTypeDef *hpcd)
    - OTG_FS_IRQHandler()
    - USB_EPStartXfer(USB_OTG_GlobalTypeDef *USBx, USB_OTG_EPTypeDef *ep, uint8_t *pTxFifo)
    - HAL_PCD_EP_Receive(PCD_HandleTypeDef *hpcd, uint8_t *pTxFifo, uint32_t *pTxFifo)
    - USB0_LL_PrepereReceive(USB0_HandleTypeDef *hpb, uint8_t *pTxFifo, uint16_t *pTxFifo)
    - HAL_PCD_EP_Transmit(PCD_HandleTypeDef *hpcd, uint8_t *pTxFifo, uint32_t *pTxFifo)
    - USB0_LL_Transmit(USB0_HandleTypeDef *hpb, uint8_t *pTxFifo, uint16_t *pTxFifo)
    - USB_HC_StartXfer(USB_OTG_GlobalTypeDef *USBx, USB_OTG_HCTypeDef *hcb, uint8_t *pTxFifo)
```

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelDE)

Highlight Inactive Code

```

157
158 #if defined ( __ICARM__ ) /* IAR Compiler */
159 #pragma data_alignment=4
160 #endif /* defined ( __ICARM__ ) */
161 /** USB standard device descriptor. */
162 __ALIGN_BEGIN uint8_t USB_FS_DeviceDesc(USB_LEN_DEV_DESC) __ALIGN_END =
163 {
164     0x12, /*bLength */
165     USB_DESC_TYPE_DEVICE, /*bDescriptorType*/
166     #if (USB_LPM_ENABLED == 1)
167     0x01, /*bcdUSB */ /* changed to USB version 2.01
168                                     in order to support LPM L1 suspend
169                                     resume test of USBVC3.0*/
170 #else
171     0x00, /*bcdUSB */
172 #endif /* (USB_LPM_ENABLED == 1) */
173     0x02,
174     0x00, /*bDeviceClass*/
175     0x00, /*bDeviceSubClass*/
176     0x00, /*bDeviceProtocol*/
177     USB_MAX_EP0_SIZE, /*bMaxPacketSize*/
178     LOBYTE(USB0_VID), /*idVendor*/
179     HIBYTE(USB0_VID), /*idVendor*/

```

Auto-Complete

```

97  /* USER CODE BEGIN 2 */
98  HAL_GPIO_I
99  /* USER CO
100
101  /* Inifinit
102  /* USER CO
103  while (1)
104  {
105  /* USER
106
107  /* USER
108  }
109  /* USER CO
110  }
111
112  /**
113   * @brief S
114   * @retval

```

File Diff/Compare

Compare (F746-DISCO-TEST/Src/main.c - F746-DISCO-TEST-2/Src/main.c)

C Compare

- StartDefaultTask
- SystemClock_Config
- cmsis_os.h
- fatfs.h
- usb_device.h

C Compare (Cannot Compare Structures)

C Compare Viewer

F746-DISCO-TEST/Src/main.c

```

20
21/* Includes -----
22#include "main.h"
23#include "usb_device.h"
24
25/* Private includes -----
26/* USER CODE BEGIN Includes */
27
28/* USER CODE END Includes */
29

```

F746-DISCO-TEST-2/Src/main.c

```

20
21/* Includes -----
22#include "main.h"
23#include "cmsis_os.h"
24#include "fatfs.h"
25#include "usb_host.h"
26
27/* Private includes -----
28/* USER CODE BEGIN Includes */
29

```

Syntax Highlight

```

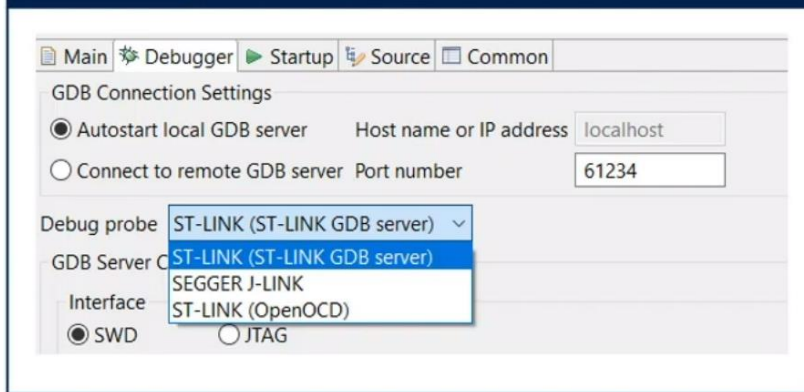
116=void SystemClockConfig(void)
117 {
118     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
119     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
120     RCC_PeriphCLKInitTypeDef PeriphClkInitStruct = {0};
121
122     /** Configure the main internal regulator output voltage
123     */
124     __HAL_RCC_PWR_CLK_ENABLE();
125     __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE3);
126     /** Initializes the CPU, AHB and APB buses clocks
127     */
128     RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI|RCC_OSCILLAT
129     RCC_OscInitStruct.HSEState = RCC_HSE_ON;
130     RCC_OscInitStruct.HSISetup = RCC_HSI_ON;
131     RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
132     RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
133     RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
134     RCC_OscInitStruct.PLL.PLLM = 15;
135     RCC_OscInitStruct.PLL.PLLN = 144;

```

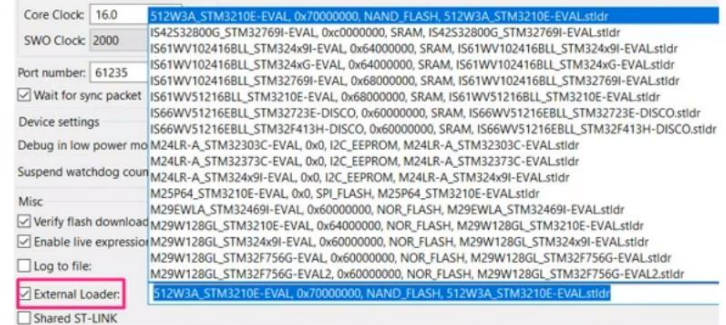

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubeIDE)

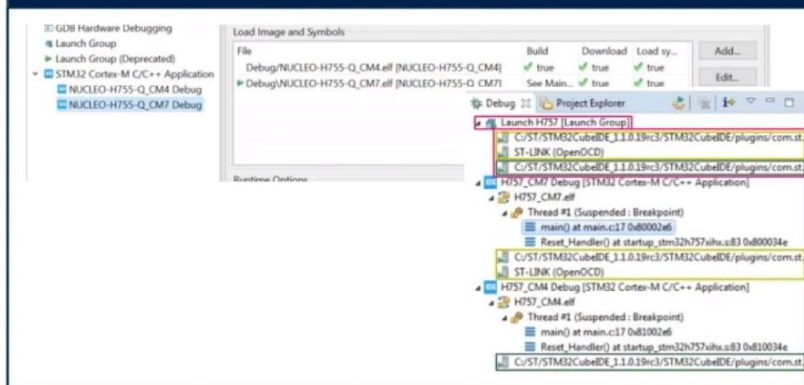
Open OCD or GDB server



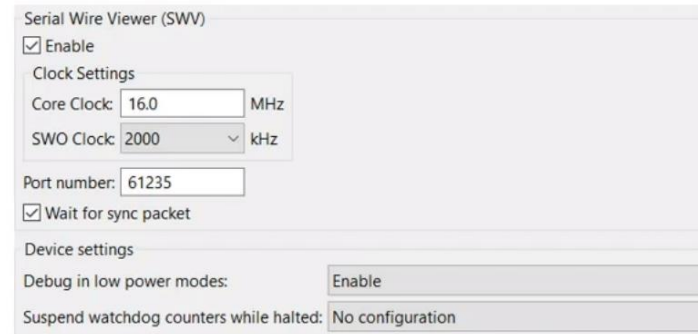
Program your external Memory



Multicore Debug







Trace Support



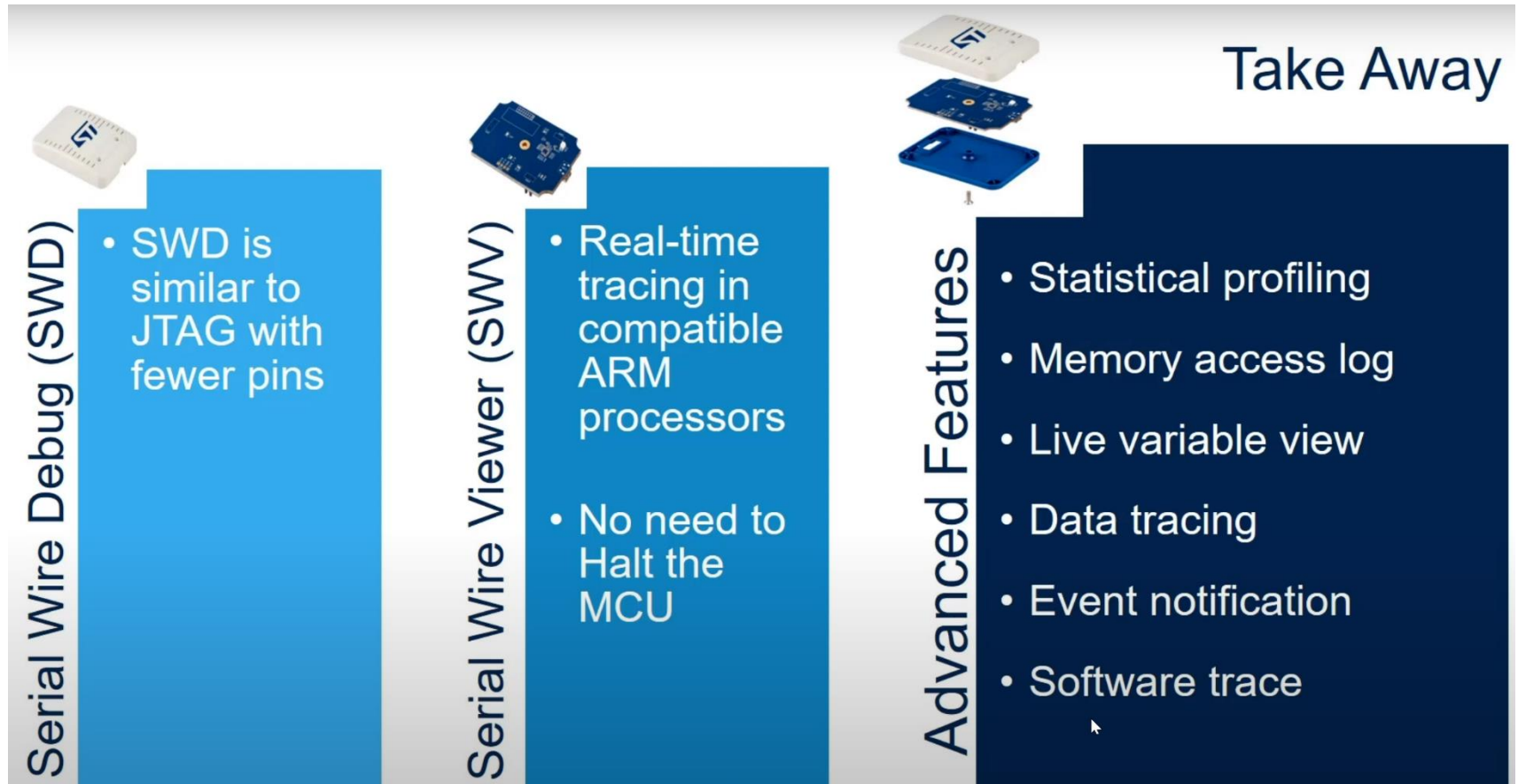
2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubeIDE)

				
Core	Cortex M0 / M0+	Cortex M3 / M4	Cortex M33	Cortex M7
Debug	SWD*	JTAG / SWD	JTAG / SWD	JTAG / SWD
Trace	No	Trace port Serial Wire Viewer	Trace port Serial Wire Viewer	Trace port Serial Wire Viewer

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubeIDE)



2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelDE)

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelDE)

2. Programiranje vgrajenih sistemov

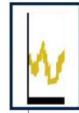
Razvoj in razhroščevanje (primer CubelIDE)



Using Data Trace
& Live Watch



Timing
Measurement



Using Data Trace
Timeline Graph



Exception Log &
Timeline Graph



printf()
redirection



Statistical Profiling

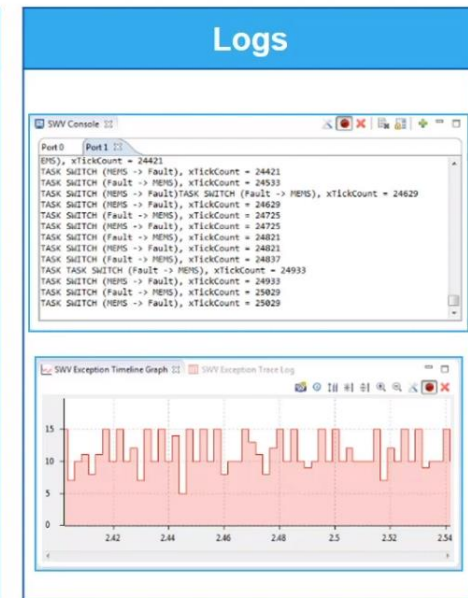
• SWV adds:

Real Time Trace

- That uses the SWD port and the SWO pin.

Advanced Debugging

- Without Halting the MCU



2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelDE)

2. Programiranje vgrajenih sistemov

Razvoj in razhroščevanje (primer CubelDE)

2. Programiranje vgrajenih sistemov

Kaj po koncu razvoja ?

- ▶ Dokumentacija (!*?)
- ▶ Spremljanje delovanja (kurativa) :
 - ▶ serijska konzola
 - ▶ log datoteke, sporočanje napak, daljinski nadzor
- ▶ primer kritične napake:

```
2015-01-11 05:45:37 CRIT 232 0 FP WDT has expired
2015-01-11 05:46:21 CRIT 232 0 MNG WDT has expired
```

- ▶ primer pomembne napake, ki zahteva popravke v kodi :

```
2015-01-09 15:00:02 INFO 60 0 CMDEXECUTE CMD:Execute Cmd[72]
2015-01-09 15:00:02 INFO 60 0 CMDEXECUTE CMD:SendSett
2015-01-09 15:04:02 CRIT 232 0 CMDEXECUTE WDT has expired
```



spremljanje

3. Nivoji programiranja – jeziki, knjižnice

Baremetal - zbirnik

```
INIT_IO:
push {r5, r6, lr}
// Enable GPIO Peripheral Clock (bit 3 in AHB1ENR register)
ldr r6, =RCC_AHB1ENR // Load peripheral clock reg address to r6
ldr r5, [r6] // Read its content to r5
orr r5, 0x00000008 // Set bit 3 to enable GPIO clock
str r5, [r6] // Store result in peripheral clock register

// Make GPIO Pin12 as output pin (bits 25:24 in MODER register)
ldr r6, =GPIO_BASE // Load GPIO BASE address to r6
ldr r5, [r6, #GPIO_MODER] // Read GPIO_MODER content to r5
and r5, 0x00FFFFFF // Clear bits 31-24 for P12-15
orr r5, 0x55000000 // Write 01 to bits 31-24 for P12-15
str r5, [r6] // Store result in GPIO MODER register
pop {r5, r6, pc}
```

```
LED_ON:
push {r5, r6, lr}
// Set GPIO Pins to 1 (through BSSR register)
ldr r6, =GPIO_BASE // Load GPIO BASE address to r6
mov r5, #LEDs_ON
str r5, [r6, #GPIO_BSSR] // Write to BSSR register
pop {r5, r6, pc}
```

```
LED_OFF:
push {r5, r6, lr}
// Set GPIO Pins to 0 (through BSSR register)
ldr r6, =GPIO_BASE // Load GPIO BASE address to r6
mov r5, #LEDs_OFF
str r5, [r6, #GPIO_BSSR] // Write to BSSR register
pop {r5, r6, pc}
```

https://github.com/LAPSyLAB/ORLab-STIM32/tree/main/GPIO_LEDs

https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/LED_GPIO_C_Baremetal_C

Baremetal - C

```
/* USER CODE BEGIN 2 */

RCC->AHB1ENR |= 0x08;
// Enable clock for GPIO
GPIO->MODER |= 0x01000000; //
MODE Register: bit 12 == out

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    GPIO->ODR ^= 0x1000; //
    Toggle PD12

/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
for (int i=0; i<0x1000000; i++) {};
// waste some time
}
/* USER CODE END 3 */
```

HAL - C

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_GPIO_TogglePin(GPIO, GPIO_PIN_12);

/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
HAL_Delay(1000);
}
/* USER CODE END 3 */

void HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx,
uint16_t GPIO_Pin)
{
    uint32_t odr;

/* Check the parameters */
assert_param(IS_GPIO_PIN(GPIO_Pin));

/* get current Output Data Register value
*/
odr = GPIOx->ODR;

/* Set selected pins that were at low
level, and reset ones that were high */
GPIOx->BSRR = ((odr & GPIO_Pin) <<
GPIO_NUMBER) | (~odr & GPIO_Pin);
}
```

https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/LED_Blink_Demo

3. Nivoji programiranja – koncepti

Ena zanka

```
{ ...  
    if (Timer_1sec) {  
        readSensors(&data);  
        send_data(&data);  
        Timer_1sec = 0;  
    }  
  
    if (Timer_50msec) {  
        readKeys(&keys);  
        readInputs(&inputs);  
        Timer_50msec = 0;  
    }  
}
```

Končni avtomat

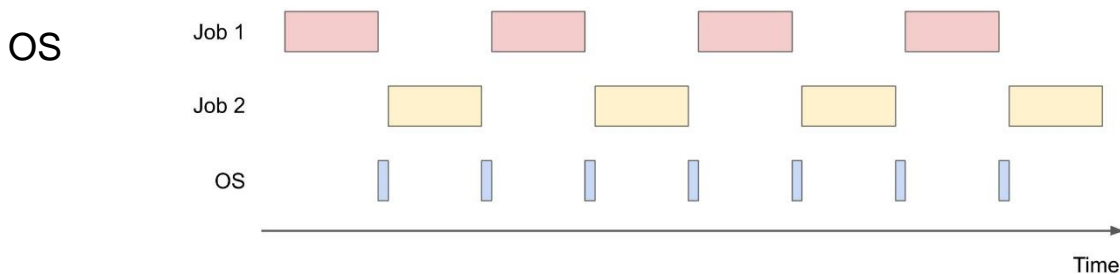
```
switch (FSM.State) {  
    case CHECK_REASON:  
        ///  
        FSM.State: after reset.  
  
        if VSE_OK then  
            FSM.State = CHECK_BAUDRATE  
  
            break;  
  
    case CHECK_BAUDRATE:  
        ///  
        FSM.State: after reset.  
        ...  
  
        break;
```

RTOS

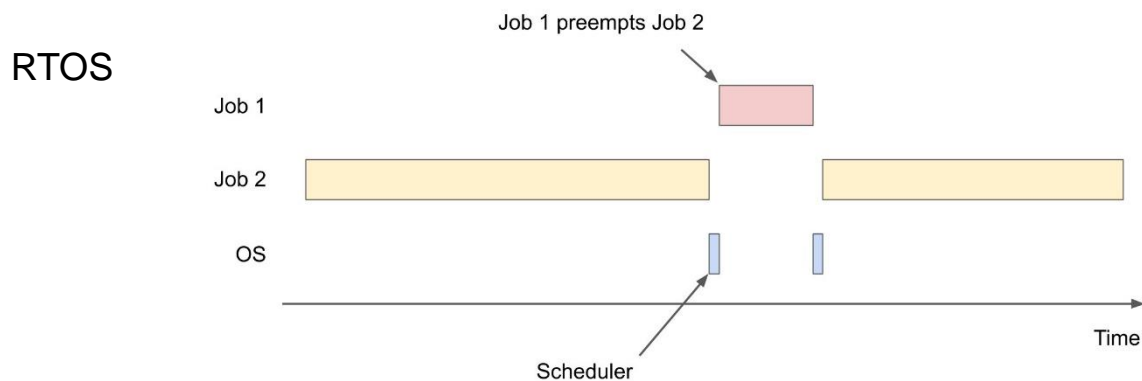
```
void StartTask02(void *argument)  
{  
    /* USER CODE BEGIN StartTask02 */  
    /* Infinite loop */  
    for(;;)  
    {  
        HAL_GPIO_TogglePin(GPIOD,  
        GPIO_PIN_13);  
        osDelay(1000);  
    }  
    /* USER CODE END StartTask02 */  
}  
  
void StartTask01(void *argument)  
{  
    /* USER CODE BEGIN StartTask01 */  
    /* Infinite loop */  
    for(;;)  
    {  
        HAL_GPIO_TogglePin(GPIOD,  
        GPIO_PIN_12);  
  
        osDelay(1000);  
    }  
    /* USER CODE END StartTask01 */  
}
```


4. Programiranje vgrajenih sistemov – OS, RTOS

General Purpose Operating System



Real-Time Operating System (RTOS)



4.1 Splošno o RTOS - Real Time Operating System

- ▶ RTOS upravlja čas in procese na mikroprocesorju ali mikrokrmilniku
 - ▶ v bistvu: „poenostavljen operacijski sistem“

- ▶ Funkcionalnosti RTOS:
 - ❑ **Večopravilnost** (multi-tasking)
 - ❑ Dodeljevanje opravil CPE s prioritetami
 - ❑ **Sinhronizacija** dostopov do virov:
 - ❑ V/I naprav
 - ❑ Pomnilnika (podatkovnih struktur)
 - ❑ **Komunikacija** med procesi (Inter-task communication)
 - ❑ Časovna predvidljivost (realno-časna odzivnost)
 - ❑ Servisiranje prekinitev

Zakaj uporabiti RTOS?

- ▶ Uporaba V/I naprav (že pripravljeni **driverji** (TCP, ETH, CANBUS,...))
- ▶ **Se splača** vse razviti iz nič (npr. svoj dodeljevalnik) ?
 - ▶ Diploma : Fabčič – 2021 – lasten RTOS „from scratch“
- ▶ Večopravilnost z možnostjo sinhronizacije
- ▶ **Prenosljivost** kode na druge CPE
- ▶ **Upravljanje z viri**
- ▶ Možnost dopolnitve z lastnimi funkcijami
- ▶ **Obstoječa podpora** za nekatere razširjene protokole:
 - TCP/IP, USB, Flash Systems, Web Servers,
 - CAN protocols, **GUI**, **SSL**, SNMP

Nekatere prednosti se hitro sprevržejo v težave in dodatno delo...

RTOS - Opravila

- ▶ Sistem oz. aplikacija je sestavljena iz več opravil
- ▶ Opravila se izmenjaje izvajajo
- ▶ V nekem trenutku je aktivno natanko eno opravilo (se izvaja na procesorju)
- ▶ RTOS odloča, kako si opravila delijo procesor („context switching“)
- ▶ Vsebina opravila („Task Context“)
 - ▶ Podatkovna struktura lastna vsakemu opravilu:
 - ▶ Vsebuje vse potrebne podatke za izvedbo opravila:
 - npr. spremenljivke, registre in sezname vseh uporabljenih virov

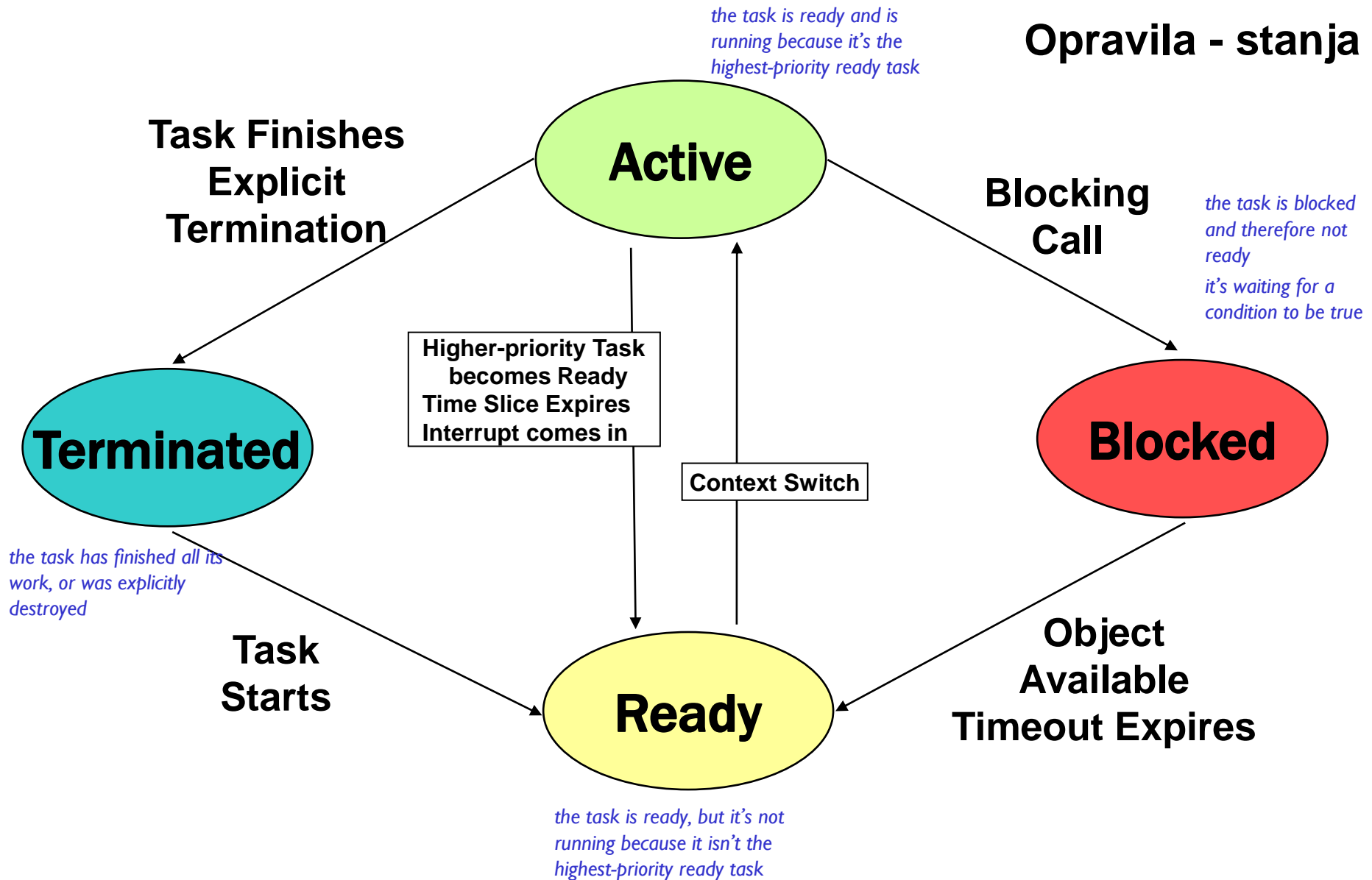
Tipična struktura kode opravila

```
void mytask(uint_32 startup_parameter) {  
    /* Task initialization code */  
    ....  
    while (1) {  
        /* Task body */  
        ....  
    }  
}
```

```
void StartTask02(void *argument)  
{  
    /* USER CODE BEGIN StartTask02 */  
    /* Infinite loop */  
    for(;;)  
    {  
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);  
        osDelay(1000);  
    }  
    /* USER CODE END StartTask02 */  
}  
  
void StartTask01(void *argument)  
{  
    /* USER CODE BEGIN StartTask01 */  
    /* Infinite loop */  
    for(;;)  
    {  
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);  
  
        osDelay(1000);  
    }  
    /* USER CODE END StartTask01 */  
}
```



Opravila - stanja



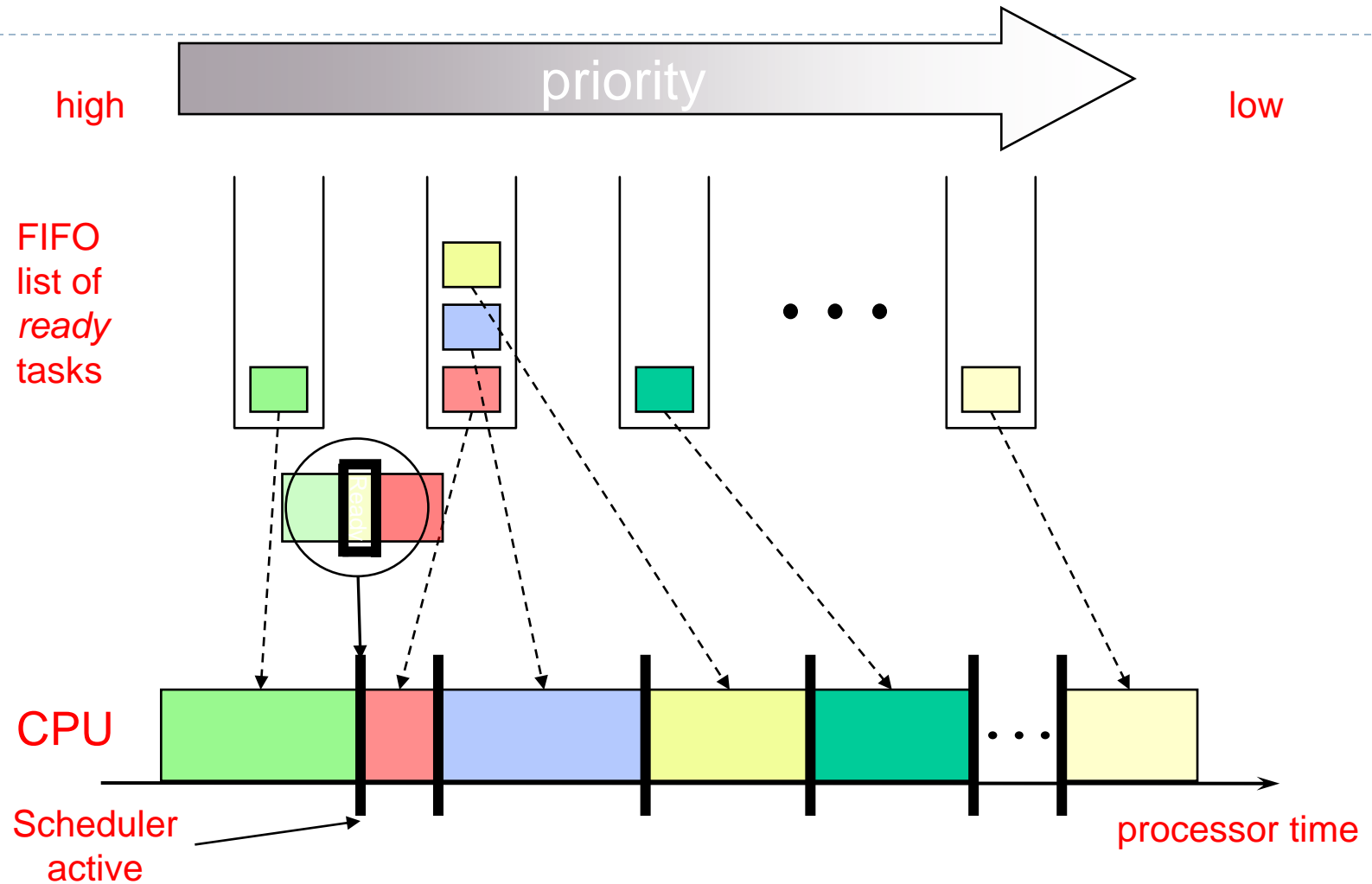
Dodeljevalnik („Scheduler“)

► Običajni načini dodeljevanja:

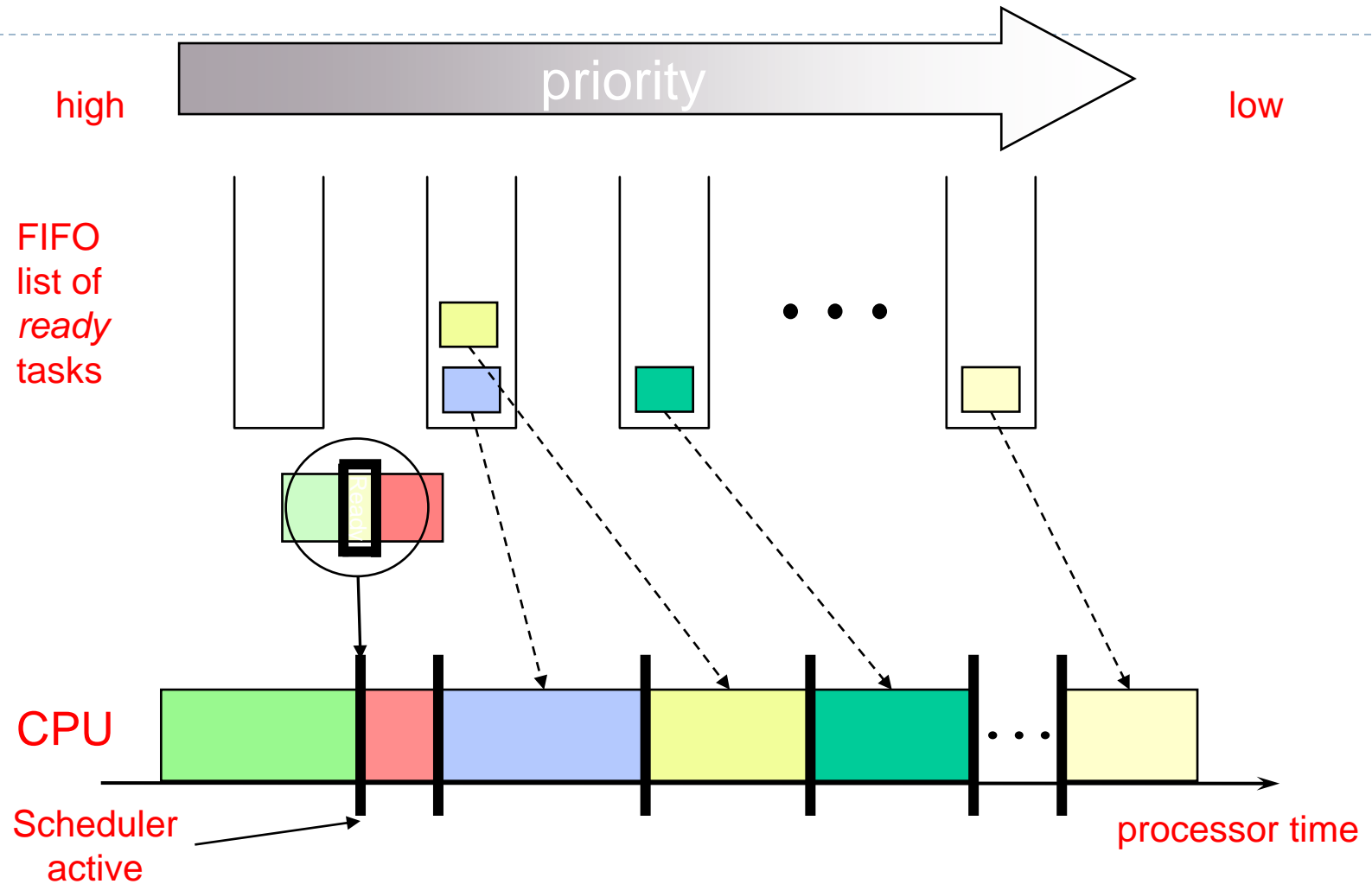
- ❑ **FIFO** (tudi „priority-based preemptive“)
 - Aktivni je tisti z najvišjo prioriteto, ki je pripravljen najdlje časa

- ❑ **Round Robin**
 - Aktivni je tisti z najvišjo prioriteto, ki je najdlje časa brez dodelitve procesorju

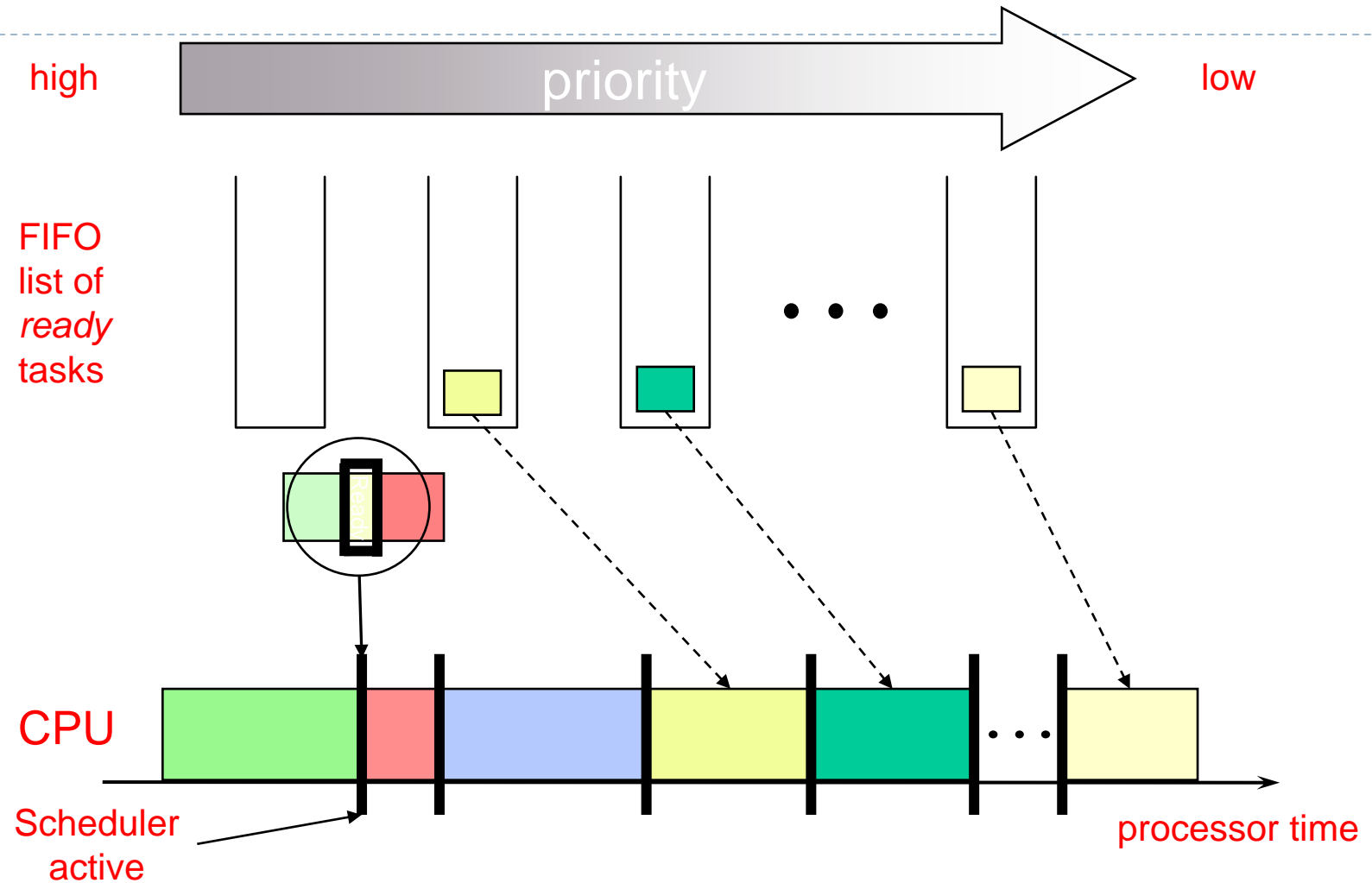
Priority Based FIFO Scheduling

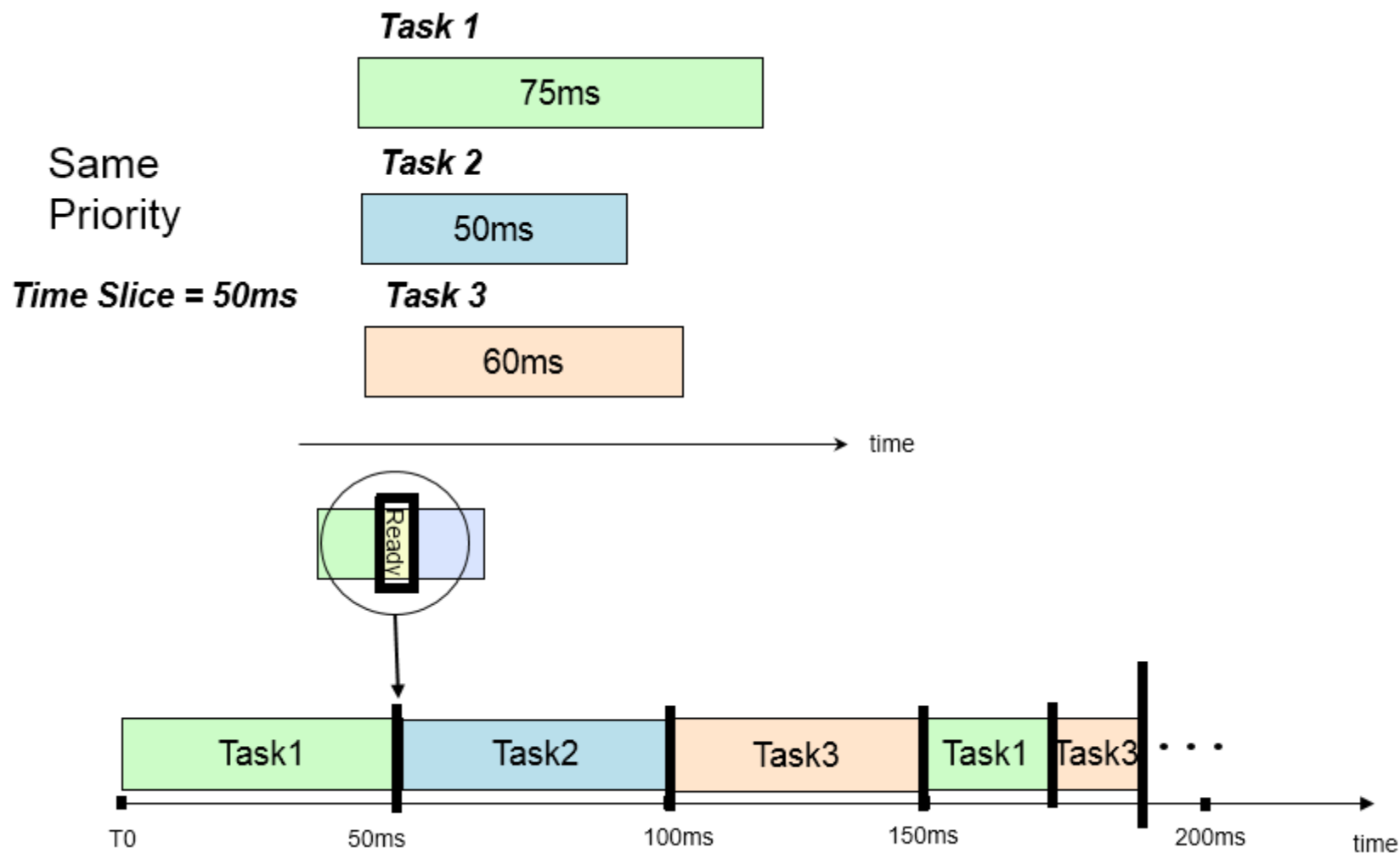


Priority Based FIFO Scheduling



Priority Based FIFO Scheduling



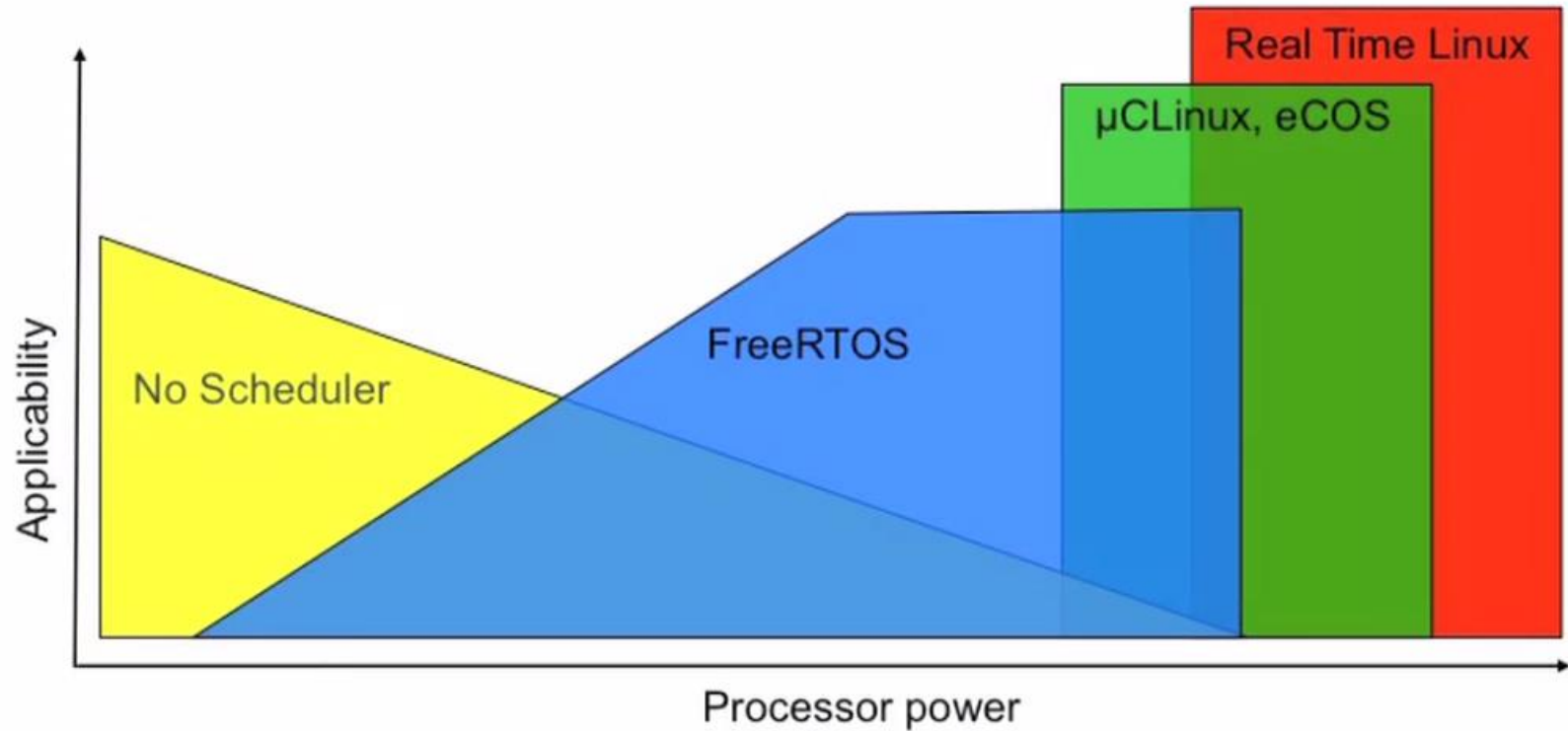


4.2. FreeRTOS (primer) :

- ▶ A Real Time Operating System
- ▶ Written by Richard Barry & FreeRTOS Team
- ▶ Huge number of users all over the world
 - ▶ 6000 Download per month
- ▶ Simple but very powerful



Kdaj uporabiti FreeRTOS ?



4.2 FreeRTOS (primer) :

Opravila („Tasks“) - Primer

```
/**
 * @brief Function implementing the ShellTask thread.
 * @param argument: Not used
 * @retval None
 */
/* USER CODE END Header_Shell_Entry */
void Shell_Entry(void const * argument)
{
    /* USER CODE BEGIN Shell_Entry */
    printf_dma ("\r\nShell Task started.\r\n");
    if ( HAL_UART_Receive_IT(&huart3, &(UARTRxBuffer[Var.Uart.RxBufferInd]), 1) != HAL_OK) {
        Error_Handler();
    }

    shell_cmd_init(); ///< Init command shell

    /* Infinite loop */
    for(;;)
    {
        shell_cmd_check_rx();    ///< check if shell character received
        osDelay(100);
        ShelluxHighWaterMark = uxTaskGetStackHighWaterMark( NULL );
    }
    /* USER CODE END Shell_Entry */
}
```

4.2. FreeRTOS (primer) :

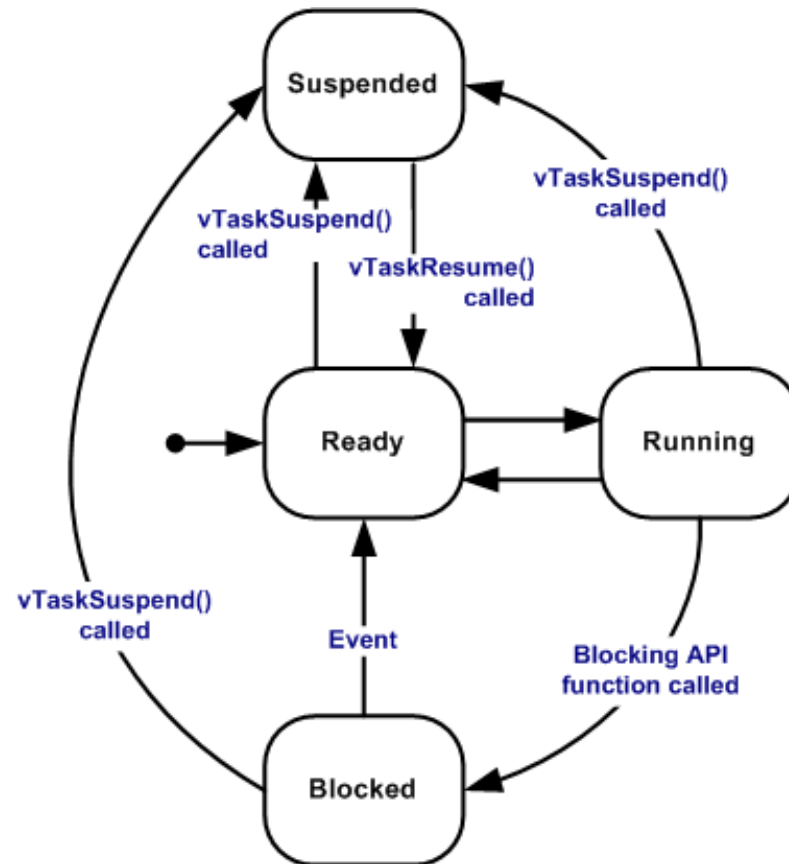
Architecture Overview

- ▶ **Tasks**
 - ▶ **task.c** , **task.h**
 - ▶ creating, scheduling, and maintaining tasks.
- ▶ **Communication**
 - ▶ **queue.c** and **queue.h** handle communication. Tasks and interrupts use queues to
 - ▶ send data to each other and
 - ▶ to signal the use of critical resources using semaphores and mutexes.
- ▶ **Hardware Interfacing**

4.2. FreeRTOS (primer) :

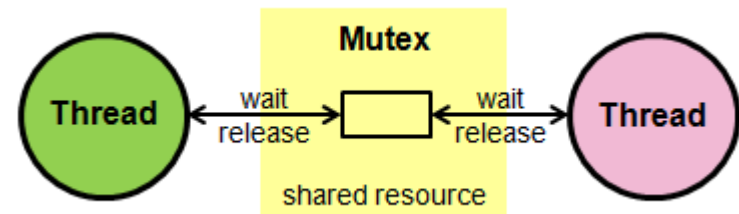
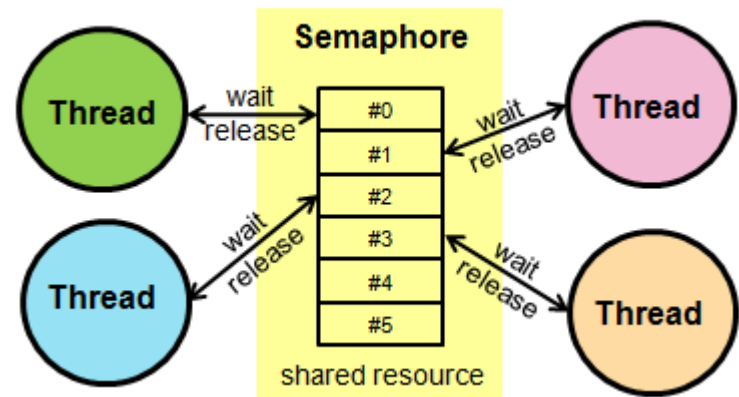
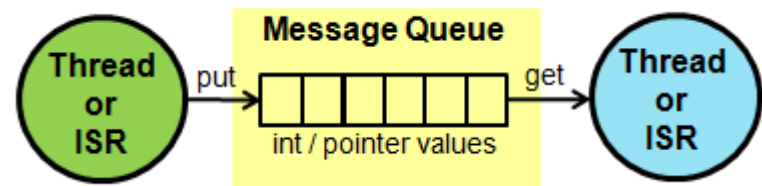
Stanje procesov

- Running
- Ready
- Blocked
- Suspended



__RTOS : Komunikacija in sinhronizacija med procesi

- ▶ Queues
- ▶ Binary Semaphores
- ▶ Counting Semaphores
- ▶ Mutexes
- ▶ Recursive Mutexes



4.2. FreeRTOS (primer) : Utripanje LED diode (vsaka v svojem procesu)

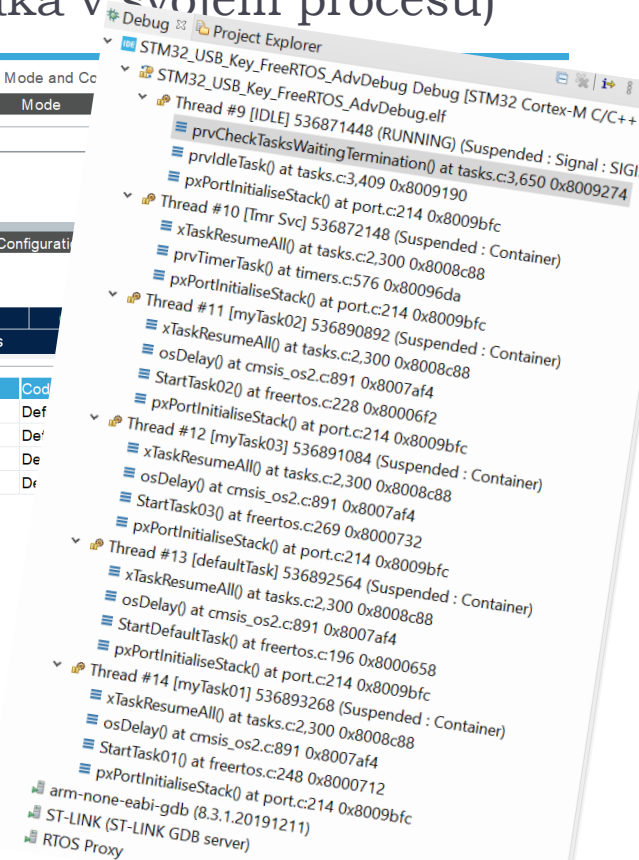
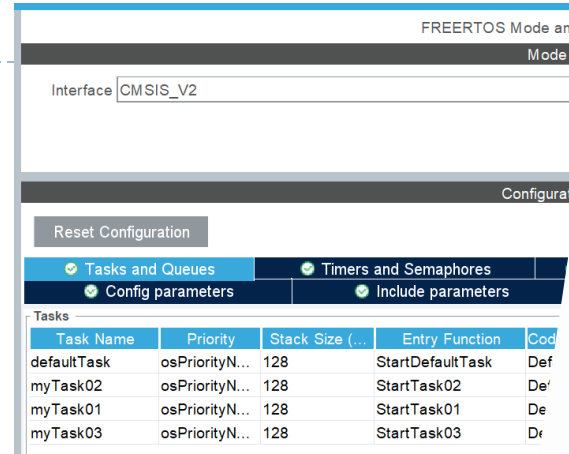
Opravila („Tasks“)

```
void StartTask02(void *argument)
{
    /* USER CODE BEGIN StartTask02 */
    /* Infinite loop */
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
        osDelay(1000);
    }
    /* USER CODE END StartTask02 */
}

void StartTask01(void *argument)
{
    /* USER CODE BEGIN StartTask01 */
    /* Infinite loop */
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);

        osDelay(1000);

    }
    /* USER CODE END StartTask01 */
}
```



Name	Priority (B...	Start of Stack	Top of Stack	State	Event Ob...	Min Free ...	Run Time...
defaultTask	24/24	0x20004f90	0x200050fc <defaultTaskBuffer+364>	DELAYED		N/A	N/A
→ IDLE	0/0	0x200002d4	0x20000474 <Idle_Stack.10878+416>	RUNNING		N/A	N/A
myTask01	24/24	0x20005294	0x2000540c <myTask01Buffer+376>	DELAYED		N/A	N/A
myTask02	24/24	0x20005810	0x20005984 <myTask02Buffer+372>	DELAYED		N/A	N/A
myTask03	24/24	0x20005554	0x200056cc <myTask03Buffer+376>	DELAYED		N/A	N/A
Tmr Svc	2/2	0x20000590	0x20000914 <Timer_Stack.10885+900>	BLOCKED	TmrQ	N/A	N/A

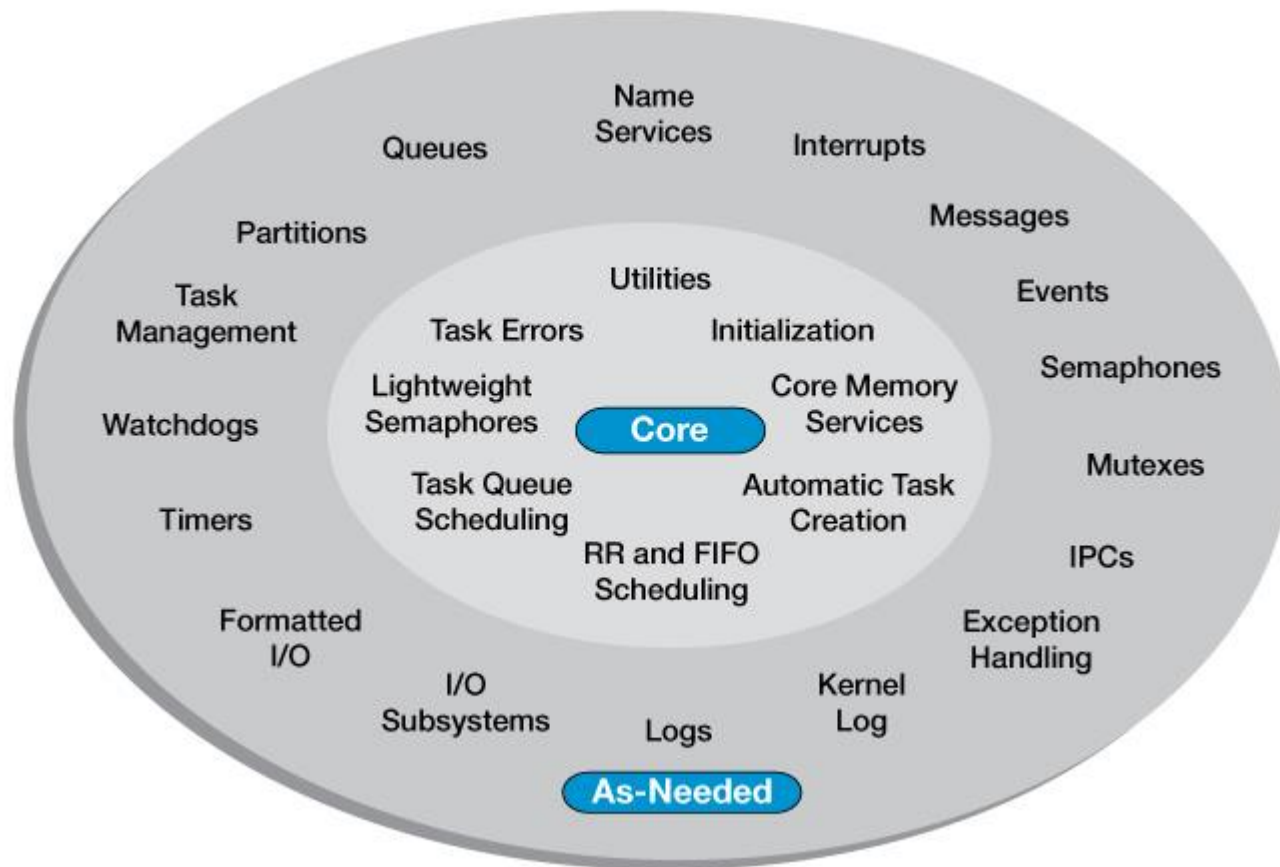


4.2. FreeRTOS (STM32F4 primer) :

Opravila („Tasks“)

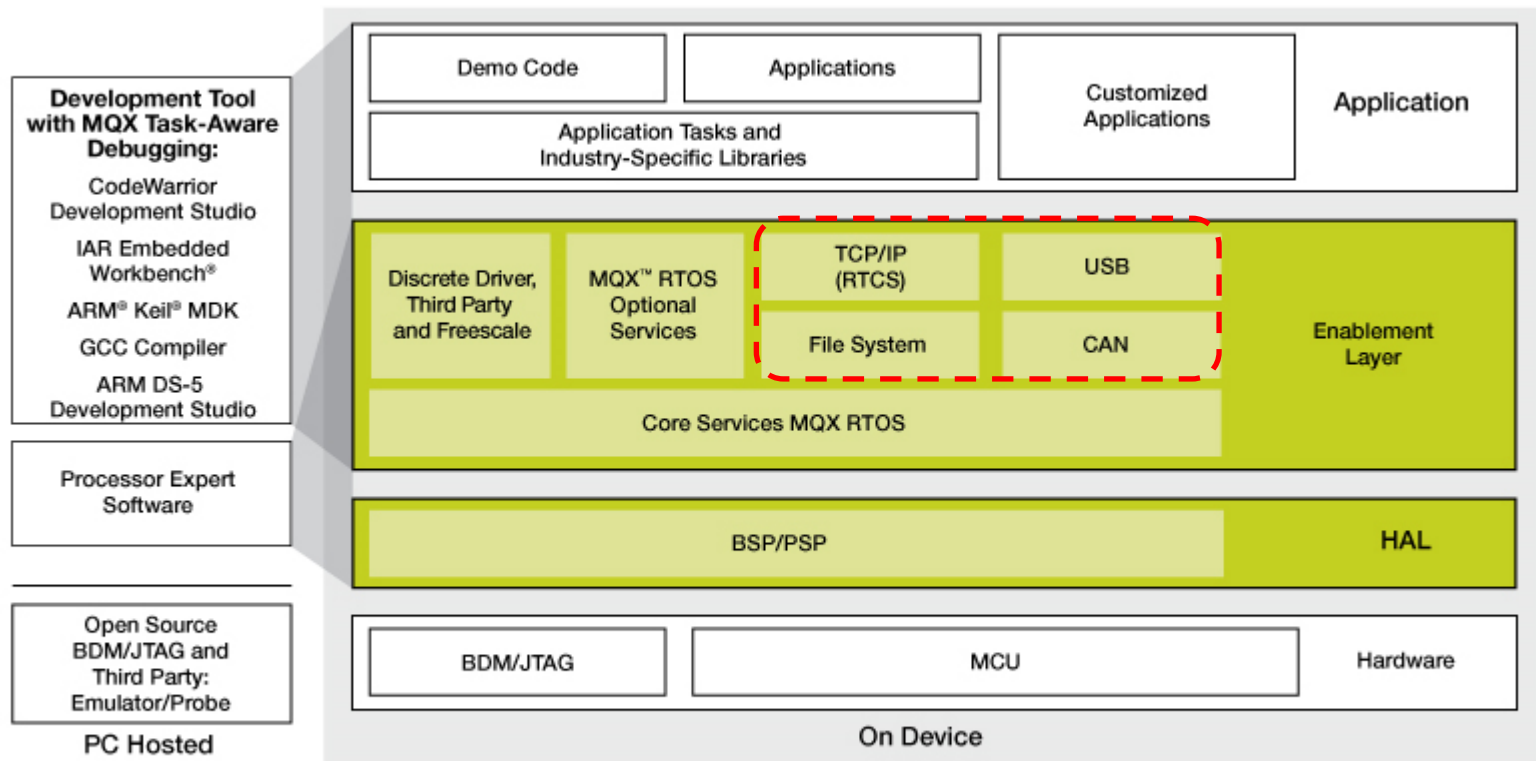
4.3 MQX RTOS (primer) :

MQX™ RTOS: Customizable Component Set



4.3 MQX RTOS (primer) :

Comprehensive Freescale Solution



 Freescale MQX™ Software Solutions

4.3 MQX RTOS (primer) :

Opravila („Tasks“)

```
const TASK_TEMPLATE_STRUCT MQX_template_list[] =
{
    /* Task Index,    Function,          Stack, Priority,          Name,                      Attributes,          Param, Time Slice */
    { MNG_TASK,      MngTask,          1200, TASK_PRIORITY_MNG_TASK, MNG_TASK_DES,      MQX_AUTO_START_TASK, 0,      0 },
    { SHELL_TASK,    ShellTask,          2000, TASK_PRIORITY_SHELL, SHELL_TASK_DES,    0,      0 },
    { FP_TASK,       FunPgmTask,         2000, TASK_PRIORITY_FP,      FP_TASK_DES,        0,      0 },
    { TNSH_TASK,     TelnetClientShell,  2000, TASK_PRIORITY_TNETSH, TNSH_TASK_DES,     0,      0 },
    { TCPCLIENT_TASK, TCPClient_Task, 2000, TASK_PRIORITY_TCPCLIENT, TCPCLIENT_TASK_DES, 0,      0 },
    { MODBUS_TASK, Modbus_Task, 2000, TASK_PRIORITY_MODBUS, MODBUS_TASK_DES, 0,      0 },
    { EVTALM_TASK, EventAlmTask, 2000, TASK_PRIORITY_EVTALM, EVTALM_TASK_DES, 0,      0 },
    { AIN_TASK,      AinTask,          500,  TASK_PRIORITY_AIN,      AIN_TASK_DES,        0,      0 },
    { NETMNG_TASK, NetMngTask, 1000, TASK_PRIORITY_NETMNG, NETMNG_TASK_DES,    0,      0 },
    { 0 }
};
```

4.3 MQX RTOS (primer MQX opravila) :

Glavna regulacijska zanka („FP_TASK“)

```
void FunPgmTask (uint_32 initial_data)
{
    FunPgmInit();

    // register task for system messages
    rc = SysMsgRegister ();

    // WDT control
    WdtRegister (15000, WDT_ACTION_LOG);

    // ----- main execution loop -----
    while (TRUE) {

        _time_get_elapsed (&fp_start_time); //Measure processing time fp_start_time

        WdtReset ();

        FunPrepareFPData();           // Prepare FP data
        FunRegulation();              // Iterate regulation loops
        FunCommitFPData();           // Commit any changes back to system

        _time_get_elapsed (&fp_end_time); //Measure processing time
        _time_diff (&fp_start_time, &fp_end_time, &fp_loop_time); // get elapsed time
        FPLoopTime=(fp_loop_time.SECONDS * 1000) + fp_loop_time.MILLISECONDS;

        _time_delay(1000-FPLoopTime); // wait for 1000 ms - loop time in ms
    }
    _task_block();                  // Shouldn't reach this point
}
```

```
/** @brief FP: Main Functional Program Task.
Calls FunPgmInit for initialization and then runs endless main FP loop.
*
* This is main functional program task.
* It will first run Initializations: FunPgmInit();
* Then it will proceed in endless loop :
*     FunPrepareFPData(); // Prepare FP data
*     FunRegulation();    // Iterate regulation loops
*     FunCommitFPData();  // Commit any changes back to system
*                         check if settings changed - if yes, then read all settings
*/
```

void FunPgmTask (uint_32 initial_data)

FP: Main Functional Program Task. Calls FunPgmInit for initialization and then runs endless main FP loop.

This is main functional program task. It will first run Initializations: **FunPgmInit()**; Then it will proceed in endless loop : **FunPrepareFPData()**; // Prepare FP data **FunRegulation()**; // Iterate regulation loops **FunCommitFPData()**; // Commit any changes back to system check if settings changed - if yes, then read all settings

Todo:

Temporary - shouldn't be used in production code !!!

Definition at line 139 of file fp.c.

References APPCFG_DEFAULT_FP_USER_ACCCODE, APPDBG_PRINTF, D13_GVARS::Day, FunCommitFPData(), FunLogCurrentState(), FunPgmInit(), FunPrepareFPData(), FunRegulation(), FunSimCommitFPData(), FunSimLogCurrentState(), FunSimPgmInit(), FunSimPrepareFPData(), FunSimRegulation(), FP_DATA::GVars, D13_GVARS::Hour, D13_GVARS::Minute, D13_GVARS::Month, Read_FPSettings(), D13_GVARS::Second, and D13_GVARS::Year.