Campo - Unterstützung #82

Add test functions for periphery

15.10.2021 12:21 - Redmine Admin

Status:	Erledigt	**Beginn:**	15.10.2021
Priorität:	Normal	**Abgabedatum:**	
Zugewiesen an:	Maximilian Seesslen	**% erledigt:**	0%
Kategorie:		**Geschätzter Aufwand:**	0.00 Stunde
Zielversion:		**Aufgewendete Zeit:**	0.00 Stunde
CS Zielversion:			

Beschreibung

The bringup software should be able to easily test periphery.

CANIO already has some test routines. They should be available in general.
Move them to a tests.cpp. No Arena stuff.

Having an test-class could help to make some overall information.

testSpiFlash(pPlatform->m_spi);
testSpiLoopback(pPlatform->m_spi);
testI2cEeprom(pPlatform->m_i2c, bottom=true);
testInternFlash( );
testCan(ECanSpeed, ECanTest::ExternLoopback/ECanTest::Mirror);

The test routines should not be in the biwak periphery clases. There will be some assumptions or depencies.
Just use functions or a seperate test class.

Those tests will also provide units-tests for biwak itself.

The fosh commands for i2c and sf are waste of space for regular use.

Historie

#1 - 08.01.2022 11:49 - Maximilian Seesslen

- Zielversion wurde auf v2.0.1 gesetzt

#2 - 10.01.2022 14:29 - Redmine Admin

- Beschreibung aktualisiert

#3 - 10.01.2022 15:03 - Redmine Admin

- Beschreibung aktualisiert

- Zielversion wurde von v2.0.1 zu v2.1.0 geändert

#4 - 10.01.2022 16:10 - Redmine Admin

- Beschreibung aktualisiert

#5 - 25.05.2022 12:30 - Maximilian Seesslen

- Beschreibung aktualisiert

#6 - 16.08.2022 17:42 - Maximilian Seesslen

An nice protocoll should be created without the interaction (e.g. "press button") output.
An test should have sub-tests. Having nested lists might be a problem for 16K-MCUs.

Tests can not be run automatically because there might be something to prepare (e.g. select-switch).

```
bringup.setTest("UART");
ltest("Receive", ch=='y', ch);
```

```
CList<CResult> m_results;

class CResult 
{
 const char *group;
 const char *testString;
 enum  EResult;
}

CBringup::setGroup(const char *groupString)
{

}

CBringup::printProtocoll
{
  const char *currentGroup;
  printf( "Test protocol:\n"
          "==============\n\n");
  for(CResult &result: m_results)
  {
     if( currentGroup != result->group )
     {
         currentGroup = result->group;
         printf( "%s:\n", currentGroup);
         currentGroup = result->group;
     }
     printf( "%s:\n", currentGroup);
  }
}
```

#7 - 16.08.2022 18:44 - Maximilian Seesslen

Markdown can be used for protocol;

1. Test Protocol 'CANDis'

System information	
:------	:----------
Serial number	3
Rev. Campo	0c8659898+3
Rev. Bringup	0c865
Rev. Biwak	0c865
Rev. Lepto	0c862322

UART	
:------	----------:
data	path to data files to supply the data that will be passed into templates.
engine	engine to be used for processing templates. Handlebars is the default.
ext	extension to be used for dest files.
Receive char	OK
Receive another char	Failure

CAN	
:--	:----------
Transmit message	OK
Receive message	Failure
CANPong-message-ID	Failure

USB	
:--	--:
Receie String from virtual keyboard	OK

1. Result

:--	:----------
Good	12
Warning	12
Bad	12
Total	12

#8 - 17.08.2022 15:50 - Maximilian Seesslen

- Projekt wurde von Biwak zu Campo geändert

- Zielversion v2.1.0 wurde gelöscht

#9 - 23.08.2022 12:01 - Maximilian Seesslen

- Status wurde von Neu zu Erledigt geändert