RapidPro System - I/O Subsystem

MPC565 RTLib Reference

For RTI RapidPro Control Unit Blockset

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/go/patches for software updates and patches.

Important Notice

Contents

About This Document

Contents

This document provides detailed descriptions of the C functions needed for an RapidPro I/O subsystem (slave processor MPC565) to complement a rapid control prototyping (RCP) system from dSPACE, such as MicroAutoBox II or a DS1007 modular system (PHS-bus-based system with a DS1007 processor board) to program RTI-specific Simulink S-functions or implement your control models manually via C programs (handcoding).

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

```
%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>
```

or

```
%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>
```

Documents folder A standard folder for user-specific documents.

```
%USERPROFILE%\Documents\dSPACE\<ProductName>\
<VersionNumber>
```

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

```
%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\
<ProductName>
```

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

Data Types

Where to go from here

Information in this section

TPU-Related Data Types

Where to go from here

Information in this section

dsrpcu_tpu_eng_t

Syntax

```
typedef struct(
   UInt16 NoOfTeeth,
   UInt16 NoOfGaps,
   UInt16 NoOfMissTeeth,
   Float64 Ratio,
   Float64 AngleRes,
   Float64 PeriodAngle,
   Float64 StartAngle,
   UInt16 EdgePol,
   UInt16 ExtTrigEnable,
   Float64 ExtTrigStartAngle,
   Float64 ExtTrigPeriodAngle,
   Float64 ExtTrigPulseDuration)
dsrpcu_tpu_eng_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify the engine parameters.

Description

The structure defines the parameters of the engine.

Members

NoOfTeeth Number of crankshaft wheel teeth. The range is [1 … 3600].

NoOfGaps Number of gaps in the crankshaft wheel [0 … 10].

NoOfMissTeeth Number of missing teeth in a gap [0 … 6].

Ratio Defines a factor by which the crankshaft speed is allowed to change in the next period without generating an error. Thus, the Ratio parameter defines the relationship between two consecutive period values, that is, T_p and $T_{p,previous}$:

$(T_{p,previous} / Ratio) < T_p < (T_{p,previous} * Ratio)$;

The Ratio parameter itself depends on the number of missing teeth of a gap and must fit the following range:

$(NoOfMissTeeth/16) \leq Ratio < (16/NoOfMissTeeth)$;

For details on the Ratio parameter refer to Acceleration and deceleration (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

AngleRes Defines the angle resolution (0.1°). For details refer to Prescaling on a TPU (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

StartAngle Defines the offset of the angle counter [0° … <720°].

PeriodAngle Specifies the angular range of one period which is evaluated for speed measurement. The range is [1.0° … 120°], with a resolution of 0.1°. The PeriodAngle should be an integer multiple N of the teeth spacing:

```
PeriodAngle = N * 360° / No. of crankshaft teeth
```

EdgePol Defines the matching edge of the crankshaft signal:

- DSRPCU_TPU_CRANK_RISING_EDGE
- DSRPCU_TPU_CRANK_FALLING_EDGE

Note

The polarity depends on the configuration of the SC modules used and the configuration settings in ConfigurationDesk for RapidPro. It is not allowed to invert the crankshaft signals.

ExtTrigEnable Defines whether an external trigger signal is generated (Bit I/O channel 1):

- DSRPCU_TPU_CRANK_EXT_TRIGGER_ENABLE
- DSRPCU_TPU_CRANK_EXT_TRIGGER_DISABLE

ExtTrigStartAngle Defines the start angle of the first trigger pulse. Range is [0° ... 719.9°], resolution is 0.1°.

Note

ExtTrigStartAngle must be less than ExtTrigPeriodAngle.

ExtTrigPeriodAngle Defines the distance between the rising edges of subsequent pulses. Range is [0.1° ... 720.0°], resolution is 0.1°.

Note

The value must divide 720° without any remainder (Period trigger angle * N = 720°).
Depending on the period trigger angle, the engine speed must not exceed specific upper limits, when using the SC-DO 8/1 digital output module (DS1646):

- 0.1°: 5000 rpm
- 0.2°: 10000 rpm
- 0.3°: 15000 rpm
- 0.4°: 20000 rpm

If you use a module that can handle shorter pulse widths than the DS1646 you can raise the upper speed limit accordingly.

ExtTrigPulseDuration Defines the duration of a single pulse of the generated trigger signal. Range is [17.8 ns ... 4.55 µs], that is, [1/56 MHz ... 255/56 MHz], resolution is 1/56 MHz.

dsrpcu_tpu_eng_st_t

Syntax

```
typedef struct (
   UInt16 SyncState,
   UInt16 SyncLostCnt,
   UInt16 ToothTooShortErrCnt,
   UInt16 ToothTooLongErrCnt,
   UInt16 GapTooShortErrCnt,
   UInt16 GapTooLongErrCnt,
   UInt16 IllegalSetupErrCnt,
   UInt16 TimeOutErrCnt,
   UInt16 AngleDivergeErrCnt,
   UInt16 UnknownErrCnt,
   UInt16 ErroFlags)
dsrpcu_tpu_eng_st_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the status of the engine.

Description

The structure contains all the status information of the engine.

> **Note**
>
> In the following descriptions, Angle Computation Unit is abbreviated as ACU.

Members

SyncState Contains the synchronization state. The value is the same as the value returned by the **`dsrpcu_tpu_crank_pm_read`** function.

SyncLostCnt Counter: Whenever an engine control error occurs, the firmware assumes that synchronization is lost. The firmware disposes an re-synchronization and increments this counter.

ToothTooShortErrCnt Counter: The ACU detects that the last tooth was too short. The firmware disposes an re-synchronization and increments this counter.

ToothTooLongErrCnt Counter: The ACU detects that the last tooth was too long. The firmware disposes an re-synchronization and increments this counter.

GapTooShortErrCnt Counter: The ACU detects that the last gap was too short. The firmware disposes an re-synchronization and increments this counter.

GapTooLongErrCnt Counter: The ACU detects that the last gap was too long. The firmware disposes an re-synchronization and increments this counter.

IllegalSetupErrCnt Counter: The ACU detects that the setup of the ACU was not correct. The firmware disposes an re-synchronization and increments this counter.

TimeOutErrCnt Counter: Indicates an overflow of an ACU-internal time counter. This can be caused by an missing input signal or an input signal that is to slow. The firmware disposes an re-synchronization and increments this counter.

AngleDivergeErrCnt Counter: The angle counters of the TPUs that perform engine control are checked for ACU synchronicity every second revolution. If they are not synchronous, the firmware starts resynchronization and increments this counter.

UnknownErrCnt Counter that is incremented if an unknown error occurs.

ErroFlags Each time one of the above counters is incremented, a corresponding error flag is set. The flag is cleared the next time if its counter has not been re-incremented in the meantime.

dsrpcu_tpu_cam_t

Syntax

```
typedef struct(
   UInt16 VectorSize,
   Float64* CamVector[32],
   Float64 Tolerance,
   UInt16 EdgePol,
   UInt16 StartPol,
   Float64 SegmentStart,
   Float64 SegmentEnd,
   UInt16 IntEnable)
dsrpcu_tpu_cam_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify the characteristics of a camshaft wheel.

Description

The structure defines the spacing of one camshaft wheel and whether an interrupt is triggered if the phase measurement is enabled.

Members

VectorSize Size of the camshaft vector in number of elements [1…32].

CamVector Camshaft vector. Each transition (as addressed by the EdgePol parameter, see below) must be specified in this vector. Angle values must be entered in ascending order. Each angle value must fit the range [0° … <720°].

Tolerance Defines the standard tolerance of all angles specified in CamVector. Range is [0° … 180°].

EdgePol Specifies the edge polarity:

- DSRPCU_TPU_CAM_RISING_EDGE
- DSRPCU_TPU_CAM_FALLING_EDGE
- DSRPCU_TPU_CAM_EITHER_EDGE

Note

The polarity depends on the configuration of the SC modules used and the configuration settings in ConfigurationDesk for RapidPro. It is not allowed to invert the camshaft signals.

StartPol Specifies the signal polarity before the first transition:

- DSRPCU_TPU_CAM_START_LOW
- DSRPCU_TPU_CAM_START_HIGH

SegmentStart Start angle of the camshaft evaluation segment. Range is [0° … <720°]. Refer to Logging the Camshaft Signal Within the Evaluation

Segment (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮). SegmentStart must equal an integer multiple N of the PeriodAngle on page 13:

```
SegmentStart = N * PeriodAngle
```

SegmentEnd End angle of the camshaft evaluation segment. Range is [0° … <720°]. Refer to Logging the Camshaft Signal Within the Evaluation Segment (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮). SegmentEnd should equal or exceed an integer multiple N of the PeriodAngle on page 13:

```
SegmentEnd >= N * PeriodAngle
```

IntEnable Defines whether an interrupt is triggered when a new result of the phase measurement is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

If phase measurement is disabled this parameter is ignored.

dsrpcu_tpu_inj_t

Syntax

```
typedef struct(
   UInt16 PulseNumber,
   Float64 PickUpDuration,
   Float64 BoostDuration,
   Float64 StartAngleBTDC[15],
   Float64 Duration[15])
dsrpcu_tpu_inj_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify the injection pulses for one channel.

Members

PulseNumber Number of pulses to be generated [1 … 15].

PickUpDuration Time period of the first injection pre-pulse (high voltage). Range is [3.2 µs … 2 ms]. The pick-up pulse has a duration of several microseconds and is used to activate the injection nozzle.

BoostDuration Time period of the second injection pre-pulse (middle voltage). Range is [0 ... 2 ms].

StartAngleBTDC Angle position before top dead center at which the pulse generation is started. The range is [-360° … 359.9°].

Duration Duration of the injection pulse. Range is [0 … 1.0 s].

The duration of injection pulses always comprises three parts:

1. High voltage pre-pulse (pick-up)
2. Middle voltage pre-pulse (boost)
3. Residual pulse

Related topics

Basics

Specification of Injection and Ignition Pulse Patterns (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

HowTos

How to Specify Injection Pulses for a Cylinder (Initial Values) (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_tpu_ign_t

Syntax

```
typedef struct(
   UInt16 PulseNumber,
   Float64 StartAngleBTDC[15],
   Float64 EndAngleBTDC[15])
dsrpcu_tpu_ign_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify the ignition pulses for one channel.

Members

PulseNumber Number of pulses to be generated [1 … 15].

StartAngleBTDC Angle position before top dead center at which the pulse generation is started. The range is [-360° … 359.9°].

EndAngleBTDC Angle position before top dead center at which the pulse generation is stopped. The range is [-360° … 359.9°].

Related topics

HowTos

How to Specify Ignition Pulses for a Cylinder (Initial Values) (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_tpu_aabp_t

Syntax

```
typedef struct(
   UInt16 PulseNumber,
   Float64 StartAngleBTDC[15],
   Float64 EndAngleBTDC[15])
dsrpcu_tpu_aabp_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify an angle-angle based pulse pattern as required for triggering knock signal measurement, for example.

Members

PulseNumber Number of pulses to be generated [1 … 15].

StartAngleBTDC Angle position before top dead center at which the pulse generation is started. The range is [-360° … 359.9°].

EndAngleBTDC Angle position before top dead center at which the pulse generation is stopped. The range is [-360° … 359.9°].

I/O PLD-Related Data Types

Where to go from here

Information in this section

dsrpcu_eng_t

Syntax

```
typedef struct(
   UInt16 NoOfTeeth,
   UInt16 NoOfGaps,
   Float64 Ratio,
   Float64 AngleRes,
   UInt16 SpeedMeasFilterEnable,
   UInt16 SpeedMeasFilter,
   UInt16 TeethForSpeedResult,
   Float64 StartAngle,
   UInt16 EdgePol,
   UInt16 ExtTrigEnable,
   Float64 ExtTrigStartAngle,
   Float64 ExtTrigPeriodAngle,
   Float64 ExtTrigPulseDuration,
   UInt16 SpeedUpperThreshold,
   UInt16 SpeedLowerThreshold,
   UInt16 RevCrankEnable,
   Float64 PDurationFwd,
   Float64 PDurationRev,
   UInt16 CrankSignalErrorMax)
dsrpcu_eng_t
```

Include file

```
dsrpcutpu.h
```

Purpose

To specify the engine parameters.

Description

The structure defines the parameters of the engine.

Members

NoOfTeeth Number of crankshaft wheel teeth. The range is [1 … 360].

NoOfGaps Number of gaps in the crankshaft wheel [0 … 10].

Ratio Defines a factor by which the crankshaft speed is allowed to change in the next period without generating an error. Thus, the Ratio parameter defines the relationship between two consecutive period values, that is, T_p and $T_{p,previous}$:
$(T_{p,previous} / Ratio) < T_p < (T_{p,previous} * Ratio)$;
For details on the Ratio parameter refer to Acceleration and deceleration (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

AngleRes Defines the angle resolution, must be set to 0.1°. For details refer to Prescaling on a TPU (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

SpeedMeasFilterEnable Defines whether an average speed is calculated:

- DSRPCU_ADDIO_CRANK_SPEED_MEAS_FILTER_DISABLE
  The SpeedMeasFilter parameter is not evaluated but assumed as 1.
- DSRPCU_ADDIO_CRANK_SPEED_MEAS_FILTER_ENABLE

SpeedMeasFilter Defines the number of crankshaft teeth that are evaluated for the calculation of an average speed.

TeethForSpeedResult Defines the number of crankshaft teeth after which a new average speed is calculated and returned.

StartAngle Defines the offset of the angle counter [0° … <720°].

EdgePol Defines the matching edge of the crankshaft signal:

- DSRPCU_ADDIO_CRANK_RISING_EDGE
- DSRPCU_ADDIO_CRANK_FALLING_EDGE

Note

The polarity depends on the configuration of the SC modules used and the configuration settings in ConfigurationDesk for RapidPro. It is not allowed to invert the crankshaft signals.

ExtTrigEnable Defines whether an external trigger signal is generated (Bit I/O channel 1):

- DSRPCU_ADDIO_CRANK_EXT_TRIGGER_ENABLE
- DSRPCU_ADDIO_CRANK_EXT_TRIGGER_DISABLE

ExtTrigStartAngle Defines the start angle of the first trigger pulse. Range is [0° ... 719.9°], resolution is 0.1°.

Note

ExtTrigStartAngle must be less than ExtTrigPeriodAngle.

ExtTrigPeriodAngle Defines the distance between the rising edges of subsequent pulses. Range is [0.1° ... 720.0°], resolution is 0.1°.

Note

The value must divide 720° without any remainder:

```
Period trigger angle * N = 720°
```

Depending on the period trigger angle, the engine speed must not exceed specific upper limits, when using the SC-DO 8/1 digital output module (DS1646):

- 0.1°: 5000 rpm
- 0.2°: 10000 rpm
  10000 rpm is the maximum possible engine speed, if the crankshaft wheel is specified via a wheel wave table.

If you use a module that can handle shorter pulse widths than the DS1646 you can raise the upper speed limit accordingly.

ExtTrigPulseDuration Defines the duration of a single pulse of the generated trigger signal. Range is [17.8 ns ... 4.55 µs], that is, [1/56 MHz ... 255/56 MHz], resolution is 1/56 MHz.

SpeedUpperThreshold Defines a speed threshold. Range is [5 ... 10000 rpm]. If speed increases above this threshold, ignition and injection signal generation is enabled.

SpeedLowerThreshold Defines a speed threshold. Range is [5 ... 10000 rpm]. If speed decreases below this threshold, ignition and injection signal generation is disabled.

RevCrankEnable Defines whether the detection of reverse crankshaft rotation is enabled:

- DSRPCU_ADDIO_CRANK_REVERSE_ENABLE
- DSRPCU_ADDIO_CRANK_REVERSE_DISABLE

PDurationFwd Pulse width in the crankshaft signal while crankshaft rotates forwards. Range is [1 µs ... 255 µs]. This parameter is evaluated only if RevCrankEnable is set to DSRPCU_ADDIO_CRANK_REVERSE_ENABLE.

PDurationRev Pulse width in the crankshaft signal while crankshaft rotates backwards. Range is [1 µs ... 255 µs]. This parameter is evaluated only if RevCrankEnable is set to DSRPCU_ADDIO_CRANK_REVERSE_ENABLE.

CrankSignalErrorMax As soon as more crankshaft signal errors are detected in a 720° interval than specified by this parameter, engine control leaves the synchronized state and starts resynchronization. The default value is DSRPCU_ADDIO_CRANK_SIGNAL_ERROR_MAX.

dsrpcu_eng_st_t

Syntax

```
typedef struct (
   UInt16 SyncState,
   UInt16 SyncLostCnt,
   UInt16 NoSyncAfter720Cnt,
   UInt16 CrankSpeedAboveUpperLimitCnt,
   UInt16 CrankEventErrorCnt,
   UInt16 CamEventErrorCnt,
   UInt16 AngleDivergeErrCnt,
   UInt16 UnknownErrCnt,
   UInt16 ErroFlags)
dsrpcu_eng_st_t
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To read the status of the engine.

Description

The structure contains all the status information of the engine.

Note

In the following descriptions, Angle Computation Unit is abbreviated as ACU.

Members

SyncState Contains the synchronization state. The value is the same as the value returned by the **dsrpcu_crank_read** function.

SyncLostCnt Counter: Whenever an engine control error occurs, the firmware assumes that synchronization is lost. The firmware disposes an re-synchronization and increments this counter.

NoSyncAfter720Cnt Counter: When engine control recognizes that no synchronization has taken place during a complete engine cycle, this counter is incremented.

CrankSpeedAboveUpperLimitCnt Counter: When engine speed exceeds the maximum speed limit defined by DSRCPU_ADDIO_CRANK_MAX_SPEED_LIMIT, this counter is incremented and engine control switches to an error state.

CrankEventErrorCnt Counter: When an unexpected edge is detected at the crankshaft input, this counter is incremented.

CamEventErrorCnt Counter: Two different events increment this counter:

- During synchronization process:

  Each unexpected or missing edge which is detected at one of the initialized camshaft signal inputs increments this counter by 1. In addition, engine control switches to an error state.

- After successful synchronization:

  Each time, no camshaft signal transition is detected within a 720° interval, this counter is incremented. This error has no influence on the synchronization state. This error is also indicated via value 4 at the **Status** port of the corresponding RPCU_CAM_TPU_BLx block.

AngleDivergeErrCnt Counter: The angle counters of the TPUs that perform engine control are checked for ACU synchronicity every second revolution. If they are not synchronous, the firmware starts resynchronization and increments this counter.

UnknownErrCnt Counter that is incremented if an unknown error occurs.

ErroFlags Each time one of the above counters is incremented, a corresponding error flag is set. The flag is cleared the next time if its counter has not been re-incremented in the meantime.

dsrpcu_cam_t

Syntax

```
typedef struct(
   UInt16 VectorSize,
   Float64 CamVector[32],
   Float64 Tolerance,
   UInt16 EdgePol,
   UInt16 StartPol,
   UInt16 IntEnable)
```

```
dsrpcu_cam_t
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To specify the characteristics of a camshaft wheel.

Description

The structure defines the spacing of one camshaft wheel and whether an interrupt is triggered if the phase measurement is enabled.

Members

VectorSize Size of the camshaft vector in number of elements [1…32].

CamVector Camshaft vector. Each transition (as addressed by the EdgePol parameter, see below) must be specified in this vector. Angle values must be entered in ascending order. Each angle value must fit the range [0° … <720°].

Tolerance Defines the standard tolerance of all angles specified in CamVector. Range is [0° … 180°].

EdgePol Specifies the edge polarity:

- DSRPCU_ADDIO_CAM_RISING_EDGE
- DSRPCU_ADDIO_CAM_FALLING_EDGE
- DSRPCU_ADDIO_CAM_EITHER_EDGE

Note

The polarity depends on the configuration of the SC modules used and the configuration settings in ConfigurationDesk for RapidPro. It is not allowed to invert the camshaft signals.

StartPol Specifies the signal polarity before the first transition:

- DSRPCU_ADDIO_CAM_START_LOW
- DSRPCU_ADDIO_CAM_START_HIGH

IntEnable Defines whether an interrupt is triggered when a new result of the phase measurement is available:

- DSRPCU_ADDIO_INT_ENABLE
- DSRPCU_ADDIO_INT_DISABLE

If phase measurement is disabled this parameter is ignored.

Setup

Where to go from here

Information in this section

Board Initialization

Where to go from here

Information in this section

dsrpcu_init

Syntax

```
void dsrpcu_init(
   dsrpcu_access_t** AccessPtr,
   dsrpcu_base_addr_t BaseAddr,
   UInt32 ChannelNo,
   UInt16 TopologyID)
```

Include file

```
DSRPCUInit.h
```

Purpose

To initialize a RapidPro system.

Description

This function initializes a RapidPro system (slave) as follows:

1. The system checks whether RapidPro hardware is connected and resets the RapidPro hardware.
2. Establishes the Master-slave communication.
3. The TopologyID is retrieved from the slave and topology data is checked.

Note

The dsrpcu_init function is locked after the first function call.

I/O mapping

None

Parameters

AccessPtr Parameter that holds the start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. The handle is created by the dsrpcu_init function and related to a certain ECU interface channel (see below).

BaseAddr Base address of the board (DS1007) / module (MicroAutoBox II) that provides the communication with the slave.

ChannelNo ECU interface channel number. (1 … 2) for DS1007, 1 for DS1401 (MicroAutoBox II).

TopologyID Number that uniquely identifies the topology of a RapidPro system. The TopologyID is stored in the hardware topology file (`*.htf`). Range is [0 … 65535].

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7800	Error	Init: timeout during check connection.	The connection between the master and the RapidPro system failed. Possible reasons, for example: ▪ RapidPro hardware not connected ▪ Wrong ECU channel specified
0x7803	Error	Init: illegal ECU channel number specified.	The specified value is invalid. Use one of the predefined symbols to specify the ChannelNo parameter.
0x7804	Error	Init: no such module available.	The communication between the master processor and the RapidPro system failed. The ECU module was not found at the given BaseAddr. Make sure, that you specified the right BaseAddr and that the specified board provides an ECU interface with a LVDS link.
0x7805	Error	Init: memory allocation error.	The memory allocation for internal data storage failed.
0x7806	Error	Init: no RPCU board connected.	The connection to the RapidPro Control Unit failed. The RapidPro system must contain a Control Unit on the first layer, connected to the master via LVDS link using an ECU interface.
0x7809	Error	Init: unequal TopologyID for RapidPro system send by slave.	The specified value is invalid. The topology ID of the specified hardware topology file (*.hwt) does not match to the connected RapidPro system.
0x7900	Error	MSC[2)]: memory allocation error.	The memory allocation for internal data storage failed.
0x7901	Error	MSC[2)]: timeout during master slave communication.	Contact dSPACE Support.
0x7902	Error	MSC[2)]: illegal version type specified.	Contact dSPACE Support.
101	Error	ECU TP1 Subinterrupt receiver init: no such module at position %d.	The connection between the RapidPro Control Unit and MicroAutoBox II failed. The MicroAutoBox II is not properly connected to the RapidPro Control Unit. Check the connection between these two.

ID	Type	Message[1)]	Description
102	Error	ECU TP1_M1 Subinterrupt receiver init: no interrupt found at position %d.	Connection between RapidPro Control Unit and the MicroAutoBox II: There is no interrupt associated to the module that has this module address.
103	Error	ECU TP1_M1 Subinterrupt receiver init: memory allocation error.	Connection between RapidPro Control Unit and MicroAutoBox II: The memory allocation for internal data storage failed.
107	Error	ECU TP1 Subinterrupt receiver init: no %d. ECU TP1 module available.	The communication between the RapidPro Control Unit and MicroAutoBox II failed. The specified module of the MicroAutoBox II does not provide an ECU interface with a LVDS link.
0x7801	Error	Init: ECU TP1 module number %d is not available in this MicroAutoBox II variant.	The connection between the RapidPro Control Unit and MicroAutoBox II failed. The specified module of the MicroAutoBox II does not provide an ECU interface with a LVDS link. There is no second, third, ... ECU module available on your MicroAutoBox II.
0x7802	Error	Init: No ECU TP1 at module position %d.	The connection between the RapidPro Control Unit and MicroAutoBox II failed. The ECU module was not found at the given BaseAddr.
141	Error	(0x%02lX) Memory allocation error in function ds4121_subint_bb_init()!	The memory allocation for internal data storage failed.
0x3800	Info	Init: RPCU initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.
101	Info	ECU TP1_M1 Subinterrupt receiver init: subinterrupt receiver initialized.	This message is generated if a MicroAutoBox II is connected to the RapidPro Control Unit and the subinterrupt receiver was initialized successfully.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01lX)".
[2)] MSC: Master-Slave Communication

Related topics

References

DSRPCU_ERROR_READ 34
dsrpcu_init_cmd_finished 30

dsrpcu_init_cmd_finished

Syntax

```
void dsrpcu_init_cmd_finished(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
DSRPCUInit.h
```

Purpose

To finish the initialization.

Description

This function is called at the end of the initialization. It induces the RapidPro system to finish the initialization:

- The RapidPro Control Unit switches to Execution mode
- All inport and output ports become active

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7901	Error	MSC: timeout during master slave communication.	The communication between the RCP system and the RPCU board failed.
0xFF01	Error	Slave TPU PWM Init: initialization of PWM failed.	Contact dSPACE Support.
0xFF03	Error	Slave TPU SM Init: initialization of stepper motor failed.	Contact dSPACE Support.
0xFF05	Error	Slave TPU F2D/PW2D Init: init of freq. or pulse width measurement failed.	Contact dSPACE Support.
0xFF07	Error	Slave TPU DigIn Init: initialization of digital in failed.	Contact dSPACE Support.
0xFF09	Error	Slave TPU PWM2D Init: initialization of PWM measurement failed.	Contact dSPACE Support.
0xFF0B	Error	Slave TPU Enc Init: initialization of encoder write failed.	Contact dSPACE Support.
0xFF0C	Error	Slave TPU Enc Init: initialization of encoder read failed.	Contact dSPACE Support.
0xFF0F	Error	Slave TPU MCPWM EA Init: init of multi channel PWM edge aligned failed.	Contact dSPACE Support.
0xFF11	Error	Slave TPU MCPWM CA Init: init of multi channel PWM center aligned failed.	Contact dSPACE Support.
0xFF13	Error	Slave TPU Crank Init: init of crankshaft speed measurement failed.	Contact dSPACE Support.
0xFF14	Error	Slave TPU Crank Init: initialization of engine status read failed.	Contact dSPACE Support.
0xFF17	Error	Slave TPU Cam Init: initialization of camshaft phase measurement failed.	Contact dSPACE Support.
0xFF18	Error	Slave TPU Cam Init: initialization of camshaft phase status write failed.	Contact dSPACE Support.

ID	Type	Message[1)]	Description
0xFF1B	Error	Slave TPU Inj Init: initialization of injection update failed.	Contact dSPACE Support.
0xFF1C	Error	Slave TPU Ign Init: initialization of ignition update failed.	Contact dSPACE Support.
0xFE05	Error	Slave QADC Init: initialization of update single conversion data failed.	Contact dSPACE Support.
0xFE06	Error	Slave QADC Init: initialization of read single conversion results failed.	Contact dSPACE Support.
0xFE07	Error	Slave QADC Init: initialization of update burst conversion data failed.	Contact dSPACE Support.
0xFE08	Error	Slave QADC Init: initialization of read burst conversion results failed.	Contact dSPACE Support.
0xFA01	Error	Slave MIOS F2D/PW2D Init: initialization of F2D or PW2D channel 1 failed.	Contact dSPACE Support.
0xFA02	Error	Slave MIOS F2D/PW2D Init: initialization of F2D or PW2D channel 2 failed.	Contact dSPACE Support.
0xFA04	Error	Slave MIOS PWM Init: initialization of PWM failed.	Contact dSPACE Support.
0xF901	Error	Slave Diag Init: memory allocation for diagnosis buffer failed.	Contact dSPACE Support.
0xF902	Error	Slave Diag Init: initialization of diagnosis data read failed.	Contact dSPACE Support.
0xF903	Error	Slave Diag Init: registration of diagnosis service failed.	Contact dSPACE Support.
0xF907	Error	Slave Diag Alive Init: initialization of diagnosis alive failed.	Contact dSPACE Support.
0xF801	Error	Slave Init: memory allocation of SMC handle failed.	Contact dSPACE Support.
0xF802	Error	Slave Init: memory allocation of slave info section failed.	Contact dSPACE Support.
0xF803	Error	Slave Init: illegal inventory management version found.	Contact dSPACE Support.
0xF804	Error	Slave Init: identification of LVDS module failed.	Contact dSPACE Support.
0xF805	Error	Slave Init: inventory management code word was not found.	Contact dSPACE Support.
0xF806	Error	Slave Init: illegal board ID found.	Contact dSPACE Support.
0xF807	Error	Slave Init: identification of processor module failed.	Contact dSPACE Support.
0xF808	Error	Slave Init: timeout during check connection.	Contact dSPACE Support.
0xF809	Error	Slave Init: initialization of SMC info section failed.	Contact dSPACE Support.
0xF80A	Error	Slave Init: failure during registration of protocol version.	Contact dSPACE Support.
0xF80B	Error	Slave Init: failure during registration of engineering protocol version.	Contact dSPACE Support.
0xF80C	Error	Slave Init: failure during registration of firmware version.	Contact dSPACE Support.
0xF80D	Error	Slave Init: failure during registration of user firmware version.	Contact dSPACE Support.
0xF80E	Error	Slave Init: failure during registration of custom firmware version.	Contact dSPACE Support.
0xF80F	Error	Slave Init: failure during registration of custom project number.	Contact dSPACE Support.

ID	Type	Message[1)]	Description
0xF810	Error	Slave Init: slave info section overflow.	Contact dSPACE Support.
0xF811	Error	Slave Init: timeout during synchronization.	Contact dSPACE Support.
0xF812	Error	Slave Init: memory allocation during synchronization failed.	Contact dSPACE Support.
0xF813	Error	Slave Init: error during initialization of external interrupt 5.	Contact dSPACE Support.
0xF814	Error	Slave Init: error during initialization of external interrupt 6.	Contact dSPACE Support.
0xF815	Error	Slave Init: error during initialization of external interrupt 7.	Contact dSPACE Support.
0xF816	Error	Slave Init: error during initialization of com module 1 interrupt.	Contact dSPACE Support.
0xF817	Error	Slave Init: error during initialization of com module 2 interrupt.	Contact dSPACE Support.
0xF818	Error	Slave Init: error during initialization of sub-interrupt handling.	Contact dSPACE Support.
0xF819	Error	Slave Init: initialization of dispatcher failed.	Contact dSPACE Support.
0xF81A	Error	Slave CPU Load Init: initialization of CPU load measurement failed.	Contact dSPACE Support.
0xF701	Error	Slave ADDIO Angle Int Init: initialization of angle interrupts failed.	Contact dSPACE Support.
0xF703	Error	Slave ADDIO Bit-IO Init: memory allocation of parameter sets failed.	Contact dSPACE Support.
0xF408	Error	Slave Config IM Init: inventory management is invalid.	Contact dSPACE Support.
0x790D	Error	MSC: unknown error code: 0x%lx.	An unexpected error occurred. Contact dSPACE Support after checking all initialization commands in your source code.
0x3801	Info	Init command finished: initialization finished successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

DSRPCU_ERROR_READ 34
dsrpcu_init 28

DSRPCU_ERROR_READ

Syntax	`DSRPCU_ERROR_READ (AccessPtr,SlaveErrorCode)`
Include file	`DSRPCUInit.h`
Purpose	To read the most recent slave error code.
I/O mapping	None
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **SlaveErrorCode** Variable that holds the error code most recently sent by the slave. The data type is UInt16.
Return value	None

DSRPCU_BACKGROUND

Syntax	`DSRPCU_BACKGROUND (`
	`  AccessPtr,`
	`  Connection)`

Include file	`DSRPCUInit.h`
Purpose	To perform the background service for the RapidPro system.
Description	This macro performs the background service for the RapidPro system. It checks the ECU-connection.
I/O mapping	None

Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **Connection** Parameter that describes the status of the ECU-connection. The data type is UInt16. ▪ ECU_TP1_CONNECTION_OK ▪ ECU_TP1_CONNECTION_LOST
Return value	None

CPU Load Measurement

Where to go from here

Information in this section

dsrpcu_cpu_load_init

Syntax

```
void dsrpcu_cpu_load_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrR,
   dsrpcu_param_t** ParamSetPtrW,
   Float64 MeasurePeriod,
   Float64 ShutdownThreshold)
```

Include file

```
DSRPCUInit.h
```

Purpose

To initialize the CPU load measurement on the RapidPro Control Unit.

Description

After successful initialization, the RapidPro system periodically transfers the results of the CPU load measurement from the slave to the master processor via the ECU interface.

The CPU load is measured as follows:

```
CPU_Load = 100% * (1 - Idle_time / Load_time)
```

```
   ; Load_time: Background execution time if load exists
```

```
   ; Idle_time: Background execution time if no load exists
```

CPU_Load is averaged over the MeasurePeriod. The MeasurePeriod parameter determines the duration of one measurement cycle.

When the CPU load exceeds the shut-down threshold the RapidPro system disables critical interrupts. Thus, the master-slave-communication will be maintained.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the CPU load value from the slave to the master processor. The function allocates and initializes the parameter set.

ParamSetPtrW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the ShutdownThreshold and the MeasurePeriod from the master to the slave processor. The function allocates and initializes the parameter set.

MeasurePeriod Duration of one CPU load measurement cycle. The possible range is [0.286 µs … 2 s]. A sensible range, however, is [1 ms …2 s].

ShutdownThreshold Initial shutdown threshold in percent. The possible range is [50% … 95%].

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7840	Error	CPULOAD Init: Illegal measurement period specified.	The specified value is outside the range permitted for the MeasurePeriod parameter.
0x7841	Error	CPULOAD Init: CPU load measurement has already been initialized.	You can initialize the CPU load measurement only once.
0x7842	Error	CPULOAD Init: memory allocation error.	The memory allocation for internal data storage failed.
0x7843	Error	CPULOAD Init: illegal shutdown threshold specified.	The function was called with an illegal shutdown threshold.
0x7844	Error	CPULOAD Init: wrong protocol version. Please update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x3804	Info	CPULOAD Init: CPU load measurement initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_cpu_load_read

Syntax

```
Int16 dsrpcu_cpu_load_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrR,
   UInt16* CpuLoad,
   UInt16* Status)
```

Include file

```
DSRPCUInit.h
```

Purpose

To read the most recent result of the CPU load measurement.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Start address of the allocated parameter set. The parameter set is used to transfer the results of the CPU load measurement from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

CpuLoad Address where the most recent result of the CPU load measurement (in percent) is stored.

Status Address where the current status of the CPU load measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

Related topics

References

dsrpcu_cpu_load_init....36
dsrpcu_cpu_load_update....39

dsrpcu_cpu_load_update

Syntax

```
Int16 dsrpcu_cpu_load_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrW,
   Float64 ShutdownThreshold)
```

Include file

```
DSRPCUInit.h
```

Purpose

To update the shut-down threshold for the CPU load measurement.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the ShutdownThreshold and the MeasurePeriod from the master to the slave processor. The function allocates and initializes the parameter set.

ShutdownThreshold New shut-down threshold in percent. The possible range is [50% … 95%].

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

Related topics

References

dsrpcu_cpu_load_init..... 36
dsrpcu_cpu_load_read..... 38

TPU Initialization

Where to go from here

Information in this section

dsrpcu_tpu_init

Syntax

```
void dsrpcu_tpu_init(
   dsrpcu_access_t* AccessPtr
   UInt16 Unit)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize a specific time-processing unit (TPU).

Description

This function globally initializes a specific TPU. The usage of sub-initialization functions (for example, dsrpcu_tpu_pwm_init) is prepared, and specific data structures are set up.

The dsrpcu_tpu_init function has to be called for each TPU separately.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F00	Error	TPU Init: illegal TPU unit number specified	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F06	Error	TPU: memory allocation error	The memory allocation for internal data storage failed.
0x3F00	Info	TPU Init: TPU X initialized successfully	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_prescaler_set 42
dsrpcu_tpu_start 44

dsrpcu_tpu_prescaler_set

Syntax

```
void dsrpcu_tpu_prescaler_set(
   dsrpcu_access_t* AccessPtr
   UInt16 Unit
   UInt16 TCR1Prescaler
   UInt16 TCR2ClkSource
   UInt16 TCR2Prescaler)
```

Include file

```
dsrpcutpu.h
```

Purpose

To configure a TPU.

Description

This function sets up the prescaler values of time counter registers TCR1 and TCR2, and the clock source of time counter register TCR2. The prescaler values determine the resolution of the time counter registers.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

TCR1Prescaler Prescaler value of time counter register TCR1:

Predefined Symbol	Value	Prescaled Clock	Resolution
DSRPCU_TPU_TCR1_PSCK_2	2	28 MHz	35.71 ns
DSRPCU_TPU_TCR1_PSCK_4	4	14 MHz	71.43 ns
DSRPCU_TPU_TCR1_PSCK_8	8	7 MHz	142.90 ns
DSRPCU_TPU_TCR1_PSCK_14	14	4 MHz	0.25 µs
DSRPCU_TPU_TCR1_PSCK_28	28	2 MHz	0.5 µs
DSRPCU_TPU_TCR1_PSCK_42	42	1.333 MHz	0.75 µs
DSRPCU_TPU_TCR1_PSCK_56	56	1 MHz	1 µs
DSRPCU_TPU_TCR1_PSCK_84	84	0.667 MHz	1.5 µs
DSRPCU_TPU_TCR1_PSCK_112	112	0.5 MHz	2 µs
DSRPCU_TPU_TCR1_PSCK_168	168	0.333 MHz	3 µs
DSRPCU_TPU_TCR1_PSCK_224	224	0.25 MHz	4 µs
DSRPCU_TPU_TCR1_PSCK_336	336	0.167 MHz	6 µs
DSRPCU_TPU_TCR1_PSCK_448	448	0.125 MHz	8 µs

TCR2ClkSource Clock source of time counter register TCR2:

- DSRPCU_TPU_TCR2_PIN_CLOCK_CTRL: Crankshaft signal
- DSRPCU_TPU_TCR2_PIN_GATE_CTRL: Processor

TPU_TCR2_PIN_CLOCK_CTRL must be selected if you want to perform engine control.

TCR2Prescaler Prescaler value of time counter register TCR2, not evaluated if TCR2ClkSource is set to TPU_TCR2_PIN_CLOCK_CTRL:

Predefined Symbol	Value	Prescaled Clock	Resolution
DSRPCU_TPU_TCR2_PSCK_8	8	7 MHz	142.9 ns
DSRPCU_TPU_TCR2_PSCK_16	16	3.5 MHz	285.7 ns
DSRPCU_TPU_TCR2_PSCK_24	24	2.333 MHz	428.6 ns
DSRPCU_TPU_TCR2_PSCK_32	32	1.75 MHz	571.4 ns
DSRPCU_TPU_TCR2_PSCK_56	56	1 MHz	1 µs
DSRPCU_TPU_TCR2_PSCK_64	64	0.875 MHz	1.143 µs
DSRPCU_TPU_TCR2_PSCK_120	120	0.467 MHz	2.143 µs

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F01	Error	TPU Prescaler Set: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F02	Error	TPU Prescaler Set: illegal TCR1 prescaler value specified.	The specified value is invalid. Use one of the predefined symbols to specify the TCR1Prescaler parameter.
0x7F03	Error	TPU Prescaler Set: illegal TCR2 prescaler value specified.	The specified value is invalid. Use one of the predefined symbols to specify the TCR2Prescaler parameter.
0x7F04	Error	TPU Prescaler Set: illegal pin control parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the TCR2ClkSource parameter.
0x3F01	Info	TPU Prescaler Set: prescaler of the TPU %d initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

Basics

Timer Setup (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_tpu_init....41
dsrpcu_tpu_start....44

dsrpcu_tpu_start

Syntax

```
void dsrpcu_tpu_start(
   dsrpcu_access_t* AccessPtr,
   UInt16 UnitMask)
```

Include file

```
dsrpcutpu.h
```

Purpose

To start one or more time-processing units (TPUs).

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

UnitMask Mask for selecting of one or more time-processing units. The different defines can be combined with a logical OR.

- DSRPCU_TPU_MASK_A: To select time-processing unit A.
- DSRPCU_TPU_MASK_B: To select time-processing unit B.
- DSRPCU_TPU_MASK_C: To select time-processing unit C.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F05	Error	TPU Start: illegal TPU unit number specified	The specified value is invalid. Use one of the predefined symbols to specify the UnitMask parameter.
0x3F11	Info	TPU Start: time-processing unit A started successfully	This message is generated if time-processing unit A started successfully and the application was compiled using the -DDEBUG_INIT option.
0x3F12	Info	TPU Start: time-processing unit B started successfully	This message is generated if time-processing unit B started successfully and the application was compiled using the -DDEBUG_INIT option.
0x3F13	Info	TPU Start: time-processing unit C started successfully	This message is generated if time-processing unit C started successfully and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Example

Suppose you want to start TPUs A and C:

```
dsrpcu_tpu_start(HAccess, DSRPCU_TPU_MASK_A|DSRPCU_TPU_MASK_C);
```

Related topics

References

dsrpcu_tpu_init....41
dsrpcu_tpu_prescaler_set....42

MIOS Initialization

dsrpcu_mios_init

Syntax

```
void dsrpcu_mios_init(
   dsrpcu_access_t* AccessPtr,
   UInt16 GlobalPrescaler)
```

Include file

```
dsrpcumios.h
```

Purpose

To initialize the MIOS14 module of the MPC565 processor.

Description

This function initializes the MIOS14 module of the MPC565 processor and specifies the global MIOS prescaler value.

The total prescaling factor depends on the global prescaler value and the channel-specific prescaler value:

```
Ptotal = Pglobal · Pchannel
```

Pglobal = 2 … 16 and Pchannel = 1 … 256

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

GlobalPrescaler Global prescaler value for all MIOS modules:

Predefined Symbol	Value	Prescaled Clock	Resolution
DSRPCU_MIOS_GLOBAL_PSCK_2	2	33 MHz	30 ns
DSRPCU_MIOS_GLOBAL_PSCK_3	3	22 MHz	45 ns
DSRPCU_MIOS_GLOBAL_PSCK_4	4	16.5 MHz	61 ns
DSRPCU_MIOS_GLOBAL_PSCK_5	5	13.2 MHz	76 ns
DSRPCU_MIOS_GLOBAL_PSCK_6	6	11 MHz	91 ns
DSRPCU_MIOS_GLOBAL_PSCK_7	7	9.429 MHz	106 ns
DSRPCU_MIOS_GLOBAL_PSCK_8	8	8.25 MHz	121 ns
DSRPCU_MIOS_GLOBAL_PSCK_9	9	7.333 MHz	136 ns

Predefined Symbol	Value	Prescaled Clock	Resolution
DSRPCU_MIOS_GLOBAL_PSCK_10	10	6.6 MHz	152 ns
DSRPCU_MIOS_GLOBAL_PSCK_11	11	6 MHz	167 ns
DSRPCU_MIOS_GLOBAL_PSCK_12	12	5.5 MHz	182 ns
DSRPCU_MIOS_GLOBAL_PSCK_13	13	5.077 MHz	197 ns
DSRPCU_MIOS_GLOBAL_PSCK_14	14	4.714 MHz	212 ns
DSRPCU_MIOS_GLOBAL_PSCK_15	15	4.4 MHz	227 ns
DSRPCU_MIOS_GLOBAL_PSCK_16	16	4.125 MHz	242 ns

Note

The total prescaling factor depends on the global prescaler value and the channel-specific prescaler value:

```
Ptotal = Pglobal · Pchannel
```

The prescaler values of the PWM channels are in the consecutive range 1 … 256.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x3A00	Info	MIOS Init: MIOS module initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.
0x7A0E	Error	MIOS Init: illegal global prescaler value specified.	The specified value is invalid. Use one of the predefined symbols to specify the GlobalPrescaler parameter.
0x7A0F	Error	MIOS: memory allocation error.	The memory allocation for internal data storage failed.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

Basics

Timer Setup (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

A/D Conversion

Objective

The RapidPro Control Unit RTLib provides functions that you can use for converting analog signals.

Where to go from here

Information in this section

Information in other sections

Comparison of Standard and Burst A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)
Gives information about the provided conversion configurations.

A/D Standard Conversion

Where to go from here

Information in this section

Information in other sections

Comparison of Standard and Burst A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)
Gives information about the provided conversion configurations.

dsrpcu_qadc_init2

Syntax

```
void dsrpcu_qadc_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   dsrpcu_param_t** ResultParamSetPtr,
   UInt16 Converter,
   UInt16 TriggerMode,
   UInt16 TriggerSource,
   UInt16 Delay,
   UInt16 ConvMode,
   UInt16 ChannelNo,
   UInt16* ChannelSeq,
   UInt16 ConversionNo,
   UInt16 IntEnable)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To initialize an A/D converter for single conversions.

Description

The function initializes one of the two A/D converters for single conversions. After a trigger occurs, A/D conversion is performed for all conversion channels successively. The parameter ChannelSeq on page 52 determines the processing order of the conversion channels.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer configuration data from the slave to the master processor.

ResultParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer conversion results from the slave to the master processor. The parameter set is initialized when the function is being executed.

Converter Specifies the A/D converter:

- DSRPCU_QADC_CONVERTER_A
- DSRPCU_QADC_CONVERTER_B

TriggerMode The following trigger modes are possible, refer also to A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮):

- Software trigger – A request on the master starts the conversion:
  - DSRPCU_QADC_SW_TRIGGER
- External trigger – A falling or rising edge starts the conversion:
  - DSRPCU_QADC_EXT_TRIGGER_FE (falling edge)
  - DSRPCU_QADC_EXT_TRIGGER_RE (rising edge)
- Trigger from another component of the RapidPro Control Unit – Angle-synchronized or PWM-synchronized (only TPU-PWM) trigger starts the conversion:
  - DSRPCU_QADC_SW_TRIGGER_ENGINE
  - DSRPCU_QADC_SW_TRIGGER_PWM

TriggerSource Depending on TriggerMode, this parameter specifies the following:

- If TriggerMode is set to DSRPCU_QADC_SW_TRIGGER_PWM, this parameter specifies the PWM channel (only TPU-PWM) that triggers the QADC module:
  - TPU_A: 1 … 16
  - TPU_B: 17 … 32
  - TPU_C: 33 … 48
- If TriggerMode is set to DSRPCU_QADC_SW_TRIGGER_ENGINE, this parameter specifies the index of the angle interrupt [1 … 22]. The indices [1 … 16] represent the fixed-position interrupts, and the indices [17 … 22] represent the periodic interrupts.

Delay Specifies the time interval between the trigger and start of the A/D conversion. You can use the following predefined symbols:

- DSRPCU_QADC_NO_DELAY
- DSRPCU_QADC_DELAY_STAGE1 = 45.71 µs
- DSRPCU_QADC_DELAY_STAGE2 = 91.43 µs
- DSRPCU_QADC_DELAY_STAGE3 = 182.9 µs
- DSRPCU_QADC_DELAY_STAGE4 = 365.7 µs
- DSRPCU_QADC_DELAY_STAGE5 = 731.4 µs
- DSRPCU_QADC_DELAY_STAGE6 = 1.463 µs
- DSRPCU_QADC_DELAY_STAGE7 = 2.926 ms
- DSRPCU_QADC_DELAY_STAGE8 = 5.851 ms
- DSRPCU_QADC_DELAY_STAGE9 = 11.70 ms
- DSRPCU_QADC_DELAY_STAGE10 = 23.41 ms
- DSRPCU_QADC_DELAY_STAGE11 = 46.81 ms

Disabled, if:

- TriggerMode on page 51 is set to
  DSRPCU_QADC_EXT_TRIGGER_FE,
  DSRPCU_QADC_EXT_TRIGGER_RE, or
  DSRPCU_QADC_SW_TRIGGER_ENGINE
- ConvMode on page 52 is set to DSRPCU_QADC_CONT_SCAN_MODE

ConvMode Specifies the conversion mode, refer to A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

- DSRPCU_QADC_SINGLE_SCAN_MODE (single mode). Only sensible if TriggerMode on page 51 is set to DSRPCU_QADC_SW_TRIGGER.
- DSRPCU_QADC_CONT_SCAN_MODE (continuous mode)

ChannelNo Total number of channels for which A/D conversion is performed:

- Max. 20 if ConversionNo = 1
- Max. 12 if ConversionNo != 1

ChannelSeq Pointer to an array of channels for which A/D conversion has to be performed (max. 20 channels). The order of the elements reflects the processing order. The data type of the array elements is UInt16.

Note

If you use the continuous mode, you should specify at least 10 A/D channels in the queue to avoid a task overrun on the RapidPro Control Unit.

ConversionNo Specifies the number of completed A/D conversions to be buffered on the slave before sent to the master:

- DSRPCU_QADC_CONV_NO_1 (1 conversion)
- DSRPCU_QADC_CONV_NO_2 (2 conversions)
- DSRPCU_QADC_CONV_NO_3 (3 conversions)
- DSRPCU_QADC_CONV_NO_4 (4 conversions)
- DSRPCU_QADC_CONV_NO_6 (6 conversions)
- DSRPCU_QADC_CONV_NO_8 (8 conversions)
- DSRPCU_QADC_CONV_NO_12 (12 conversions)
- DSRPCU_QADC_CONV_NO_16 (16 conversions)
- DSRPCU_QADC_CONV_NO_24 (24 conversions)
- DSRPCU_QADC_CONV_NO_32 (32 conversions)
- DSRPCU_QADC_CONV_NO_48 (48 conversions)
- DSRPCU_QADC_CONV_NO_64 (64 conversions)
- DSRPCU_QADC_CONV_NO_96 (96 conversions)
- DSRPCU_QADC_CONV_NO_128 (128 conversions)

Note

This parameter is irrelevant for continuous conversion mode.

IntEnable Specifies whether end-of-conversion interrupts (EOC interrupts) are generated and transferred to the master:

- DSRPCU_QADC_INT_ENABLE
- DSRPCU_QADC_INT_DISABLE

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7E00	Error	QADC: memory allocation error.	The memory allocation for internal data storage failed.
0x7E01	Error	QADC Init: illegal converter number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Converter parameter.
0x7E02	Error	QADC Init: illegal trigger mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the TriggerMode parameter.
0x7E03	Error	QADC Init: illegal delay interval specified.	The specified value is invalid. Use one of the predefined symbols to specify the Delay parameter.

ID	Type	Message[1)]	Description
0x7E04	Error	QADC Init: illegal conversion mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the ConvMode parameter.
0x7E05	Error	QADC Init: illegal number of ADC channel specified.	The specified value is invalid. Use one of the predefined symbols to specify the ChannelNo parameter.
0x7E06	Error	QADC Init: illegal ADC channel specified in ADC channel list.	The specified value(s) is/are invalid. The specified value is outside the range permitted for the ChannelSeq parameter.
0x7E07	Error	QADC Init: illegal trigger source specified in ADC channel list.	The specified value is invalid. The specified value is outside the range permitted for the TriggerSource parameter.
0x7E08	Error	QADC Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7E09	Error	QADC Init: wrong protocol version. Update application firmware"	The use of this functions requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7E0A	Error	QADC Init: illegal number of conversions in collection buffer specified"	The specified value is invalid. Use one of the predefined symbols to specify the ConversionNo parameter.
0x3E80	Warning	QADC Init: delayed A/D-conversion not possible for this mode.	A delay time can only be specified if the parameters are set as follows: ▪ ConvMode = DSRPCU_QADC_SINGLE_SCAN_MODE ▪ TriggerMode = DSRPCU_QADC_SW_TRIGGER or DSRPCU_QADC_SW_TRIGGER_PWM
0x3E00	Info	QADC Init: Converter %d initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Restrictions

If a converter is initialized for A/D burst conversion, it cannot be reconfigured for A/D conversion during run time, and vice versa.

Related topics

Basics

A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_qadc_start

Syntax

```
Int16 dsrpcu_qadc_start(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To start continuous A/D conversion.

Description

This function starts continuous A/D conversion if continuous mode is initialized for the converter (refer to parameter ConvMode on page 52). The A/D conversion channels are repeatedly processed in the specified order (refer to ChannelSeq on page 52). For the specification of the converter, refer to the parameter Converter on page 51.

Note

This function cannot be called in single conversion mode.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_init2 50
dsrpcu_qadc_read 58
dsrpcu_qadc_request 57
dsrpcu_qadc_stop 56

dsrpcu_qadc_stop

Syntax

```
Int16 dsrpcu_qadc_stop(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To stop continuous A/D conversion.

Description

This function stops continuous A/D conversion for a specific converter. For the specification of the converter, refer to the parameter Converter on page 51.

Note

This function cannot be called in single conversion mode.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_init2 50
dsrpcu_qadc_read 58
dsrpcu_qadc_request 57
dsrpcu_qadc_start 55

dsrpcu_qadc_request

Syntax

```
Int16 dsrpcu_qadc_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To trigger a single A/D conversion.

Description

This function triggers a single A/D conversion if single mode is enabled, refer to parameter ConvMode on page 52. Conversion is performed for all channels specified via the ChannelSeq on page 52 parameter. For the specification of the converter, refer to the parameter Converter on page 51.

Note

This function cannot be called in continuous conversion mode.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_init2 50
dsrpcu_qadc_read 58
dsrpcu_qadc_start 55
dsrpcu_qadc_stop 56

dsrpcu_qadc_read

Syntax

```
Int16 dsrpcu_qadc_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ResultParamSetPtr,
   UInt16 ReadMode,
   UInt16* ParCnt,
   Float64* Data,
   UInt16* Status)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To read the conversion results of an A/D converter.

Description

This function reads the results of a single A/D conversion and stores them in a buffer whose start address is specified by the Data parameter. For the specification of the converter, refer to the parameter Converter on page 51.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ResultParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer conversion results from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode

- DSRPCU_QADC_READ_MODE_NEW: dsrpcu_qadc_read() polls for new data from the slave. The data that was requested by dsrpcu_qadc_request() beforehand is returned. This option is only sensible if TriggerMode is set to DSRPCU_QADC_SW_TRIGGER.
- DSRPCU_QADC_READ_MODE_CURRENT: dsrpcu_qadc_read() returns the currently available data.

ParCnt Address where the total number of conversions performed is stored [1 … 20].

Data Address of the buffer where the conversion results are stored [0 … 1]. You must maintain that the buffer size is sufficient.

Status Address where the current status of the output is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_QADC_ERR_NO_SGL_CONV Converter has not been initialized for single conversions.

Related topics

References

A/D Burst Conversion

Where to go from here

Information in this section

dsrpcu_qadc_burst_init

Syntax

```
void dsrpcu_qadc_burst_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   dsrpcu_param_t** ResultParamSetPtr,
   UInt16 Converter,
   UInt16 TriggerMode,
   UInt16 TriggerSource,
   UInt16 Delay,
   UInt16 ConvMode,
   UInt16 ChannelNo,
   UInt16* ChannelSeq,
   UInt16 BurstSize,
   UInt16 IntEnable)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To initialize an A/D converter for burst conversion.

Description

The function initializes one of the two A/D converters for burst conversions. After the first trigger occurs, A/D burst conversion is performed for the first conversion channel. After the next trigger occurs, A/D burst conversion is performed for the

next conversion channel, and so on. The parameter ChannelSeq on page 62 determines the processing order of the conversion channels.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer configuration data from the slave to the master processor.

ResultParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer conversion results from the slave to the master processor. The parameter set is initialized when the function is being executed.

Converter Specifies the A/D converter:

- DSRPCU_QADC_CONVERTER_A
- DSRPCU_QADC_CONVERTER_B

TriggerMode The following trigger modes are possible, refer also to A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

- Software trigger – A request on the host PC starts the conversion:
  - DSRPCU_QADC_SW_TRIGGER
- External trigger – A falling or rising edge starts the conversion:
  - DSRPCU_QADC_EXT_TRIGGER_FE (falling edge)
  - DSRPCU_QADC_EXT_TRIGGER_RE (rising edge)
- Trigger from another component of the RapidPro Control Unit – Angle-synchronized or PWM-synchronized (only TPU-PWM) trigger starts the conversion:
  - DSRPCU_QADC_SW_TRIGGER_ENGINE
  - DSRPCU_QADC_SW_TRIGGER_PWM

TriggerSource Depending on TriggerMode, this parameter specifies the following:

- If TriggerMode is set to DSRPCU_QADC_SW_TRIGGER_PWM, this parameter specifies the PWM channel (only TPU-PWM) that triggers the QADC module:
  - TPU_A: 1 … 16
  - TPU_B: 17 … 32
  - TPU_C: 33 … 48

- If TriggerMode is set to DSRPCU_QADC_SW_TRIGGER_ENGINE, this parameter specifies the index of the angle interrupt [1 … 22]. The indices [1 … 16] represent the fixed-position interrupts, and the indices [17 … 22] represent the periodic interrupts.

Delay Specifies the time interval between trigger and start of the A/D conversion:

- DSRPCU_QADC_NO_DELAY
- DSRPCU_QADC_DELAY_STAGE1 = 45.71 µs
- DSRPCU_QADC_DELAY_STAGE2 = 91.43 µs
- DSRPCU_QADC_DELAY_STAGE3 = 182.9 µs
- DSRPCU_QADC_DELAY_STAGE4 = 365.7 µs
- DSRPCU_QADC_DELAY_STAGE5 = 731.4 µs
- DSRPCU_QADC_DELAY_STAGE6 = 1.463 µs
- DSRPCU_QADC_DELAY_STAGE7 = 2.926 ms
- DSRPCU_QADC_DELAY_STAGE8 = 5.851 ms
- DSRPCU_QADC_DELAY_STAGE9 = 11.70 ms
- DSRPCU_QADC_DELAY_STAGE10 = 23.41 ms
- DSRPCU_QADC_DELAY_STAGE11 = 46.81 ms

Disabled, if:

- TriggerMode on page 61 is set to
  DSRPCU_QADC_EXT_TRIGGER_FE,
  DSRPCU_QADC_EXT_TRIGGER_RE, or
  DSRPCU_QADC_SW_TRIGGER_ENGINE
- ConvMode on page 62 is set to DSRPCU_QADC_CONT_SCAN_MODE

ConvMode Specifies the conversion mode, refer to A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

- DSRPCU_QADC_SINGLE_SCAN_MODE (single mode). Only sensible if TriggerMode on page 61 is set to DSRPCU_QADC_SW_TRIGGER.
- DSRPCU_QADC_CONT_SCAN_MODE (continuous mode)

ChannelNo Total number of channels (max = 20) for which A/D burst conversion is performed.

ChannelSeq Pointer to an array of channels for which A/D burst conversion is performed (max. 20 channels). The order of the elements reflects the processing order. The data type of the array elements is UInt16.

BurstSize Size of a conversion burst. A conversion burst is a sequence of successive conversion results of a certain channel. The burst size is represented by a multiple of 64. The following parameter values and associated burst sizes are possible:

- DSRPCU_QADC_BURST_SIZE_64: Burst size = 1
- DSRPCU_QADC_BURST_SIZE_128: Burst size = 2
- DSRPCU_QADC_BURST_SIZE_192: Burst size = 3
- DSRPCU_QADC_BURST_SIZE_256: Burst size = 4
- DSRPCU_QADC_BURST_SIZE_320: Burst size = 5

- DSRPCU_QADC_BURST_SIZE_384: Burst size = 6
- DSRPCU_QADC_BURST_SIZE_448: Burst size = 7
- DSRPCU_QADC_BURST_SIZE_512: Burst size = 8
- DSRPCU_QADC_BURST_SIZE_576: Burst size = 9
- DSRPCU_QADC_BURST_SIZE_640: Burst size = 10

IntEnable Specifies whether end-of-conversion interrupts (EOC interrupts) are generated and transferred to the master:

- DSRPCU_QADC_INT_ENABLE
- DSRPCU_QADC_INT_DISABLE

Return value

None

Messages

The following messages are defined:

ID	Type	Message [1)]	Description
0x7E00	Error	QADC: memory allocation error	The memory allocation for internal data storage failed.
0x7E10	Error	QADC Burst Init: illegal converter number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Converter parameter.
0x7E11	Error	QADC Burst Init: illegal trigger mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the TriggerMode parameter.
0x7E12	Error	QADC Burst Init: illegal delay interval specified.	The specified value is invalid. Use one of the predefined symbols to specify the Delay parameter.
0x7E13	Error	QADC Burst Init: illegal conversion mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the ConvMode parameter.
0x7E14	Error	QADC Burst Init: illegal number of ADC channel specified.	The specified value is invalid. The specified value is outside the range permitted for the ChannelNo parameter.
0x7E15	Error	QADC Burst Init: illegal ADC channel specified in ADC channel list.	The specified value(s) is/are invalid. The specified value is outside the range permitted for the ChannelSeq parameter.
0x7E16	Error	QADC Burst Init: illegal size of burst specified.	The specified value is invalid. Use one of the predefined symbols to specify the BurstSize parameter.
0x7E17	Error	QADC Burst Init: illegal trigger source specified in ADC channel list.	The specified value is invalid. The specified value is outside the range permitted for the TriggerSource parameter.
0x7E18	Error	QADC Burst Init: illegal int enable parameter specified	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x3E81	Warning	QADC Burst Init: delayed A/D-conversion not possible in this mode.	A delay time can only be specified if the parameters are set as follows: ▪ ConvMode = DSRPCU_QADC_SINGLE_SCAN_MODE ▪ TriggerMode = DSRPCU_QADC_SW_TRIGGER or DSRPCU_QADC_SW_TRIGGER_PWM

ID	Type	Message [1]	Description
0x3E01	Info	QADC Burst Init: Converter %d initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Restrictions

- If ConvMode on page 62 is set to DSRPCU_QADC_CONT_SCAN_MODE, conversion burst can be performed only for one A/D conversion channel.
- If a converter is initialized for A/D conversion, it cannot be reconfigured for A/D burst conversion during run time, and vice versa.

Related topics

Basics

Comparison of Standard and Burst A/D Conversion (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_qadc_burst_start

Syntax

```
void dsrpcu_qadc_burst_start(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To start continuous A/D burst conversion.

Description

This function starts continuous A/D conversion if continuous mode is initialized for the converter (refer to parameter ConvMode on page 62). The A/D conversion channels are repeatedly processed in the specified order (refer to ConvMode). For the specification of the converter, refer to the parameter Converter on page 61.

Note

This function cannot be called in single conversion mode.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_burst_init....60
dsrpcu_qadc_burst_read....68
dsrpcu_qadc_burst_request....66
dsrpcu_qadc_burst_stop....65

dsrpcu_qadc_burst_stop

Syntax

```
void dsrpcu_qadc_burst_stop(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To stop continuous A/D burst conversion.

Description

This function stops continuous A/D burst conversion for a specific converter. For the specification of the converter, refer to the parameter Converter on page 61.

Note

This function cannot be called in single conversion mode.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_burst_init....60
dsrpcu_qadc_burst_read....68
dsrpcu_qadc_burst_request....66
dsrpcu_qadc_burst_start....64

dsrpcu_qadc_burst_request

Syntax

```
Int16 dsrpcu_qadc_burst_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To trigger a single A/D conversion burst.

Description

The function triggers a single A/D conversion burst for the current A/D conversion channel. After the conversion burst is completed, the converter switches to the next A/D conversion channel which is ready for the next conversion triggered. For the specification of the converter, refer to the parameter Converter on page 61.

Note

This function cannot be called in continuous conversion mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_qadc_burst_init....60
dsrpcu_qadc_burst_read....68
dsrpcu_qadc_burst_start....64
dsrpcu_qadc_burst_stop....65

dsrpcu_qadc_burst_read

Syntax

```
Int16 dsrpcu_qadc_burst_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ResultParamSetPtr,
   UInt16 ReadMode,
   UInt16* ParCnt,
   UInt16* CurChannel,
   Float64* Data,
   UInt16* Status)
```

Include file

```
DsRPCUQadc.h
```

Purpose

To read the burst conversion results of an A/D converter.

Description

This function reads the burst conversion results of a single A/D conversion (= burst channel) and stores them in a buffer whose start address is specified by the Data parameter. For the specification of the converter, refer to the parameter Converter on page 61.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ResultParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer conversion results from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode

- DSRPCU_QADC_READ_MODE_NEW: dsrpcu_qadc_burst_read() polls for new data from the slave. The data that was requested by dsrpcu_qadc_burst_request() beforehand is returned. This option is only sensible if TriggerMode is set to DSRPCU_QADC_SW_TRIGGER.
- DSRPCU_QADC_READ_MODE_CURRENT: dsrpcu_qadc_burst_read() returns the current available data.

ParCnt Address where the total number of conversions performed is stored [1 … 10]. You have to multiply this number by 64 to get the burst size.

CurChannel Index of the A/D conversion channel (see ADC channel list) for which the last burst conversion was performed [1 … 20].

Data Address of the buffer where the conversion results are stored [0 … 1]. The size of the buffer must be at least BurstSize*64.

For details on the BurstSize parameter, refer to BurstSize on page 62.

Status Address where the current status of the conversion is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_QADC_ERR_NO_BURST_CONV The converter has not been initialized for burst conversion.

Related topics

References

Bit I/O

Objective	The RapidPro Control Unit RTLib provides functions that you can use for reading/writing digital signals.
Interrupts	Processing digital signals via a TPU unit allows interrupts to be transferred to the master. This is not possible when digital signals are processed via the I/O PLD (Programmable Logic Device).
Where to go from here	Information in this section

Bit I/O via I/O PLD

Interrupts

Processing digital signals via a TPU unit allows interrupts to be transferred to the master. This is not possible when digital signals are processed via the I/O PLD (Programmable Logic Device).

Where to go from here

Information in this section

dsrpcu_bio_init2

Syntax

```
void dsrpcu_bio_init2 (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt8 GroupNo,
   UInt8 Mask,
   UInt8 Mode,
   UInt8 InitData)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To initialize the bit I/O ports.

Description

This function can be called multiple times for one bit I/O group during initialization, which allows each bit I/O port of a bit I/O group to be configured separately. This allows you, for example, to have input as well as output ports within one group, and to update the ports at different times.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer data from the slave to the master processor, or vice versa, depending on the Mode parameter. The parameter set is related to a specific bit I/O group. The parameter set is initialized when the function is being executed.

Note

If you want to address single bits of a bit I/O group, you have to call the function for each single bit. You must use a separate parameter set for each function call.

GroupNo Specifies the bit I/O group for which bit I/O ports have to be initialized:

- DSRPCU_ADDIO_BIO_GROUP1
- DSRPCU_ADDIO_BIO_GROUP2
- DSRPCU_ADDIO_BIO_GROUP3
- DSRPCU_ADDIO_BIO_GROUP4
- DSRPCU_ADDIO_BIO_GROUP5

Mask Specifies the bit I/O ports of the bit I/O group. A bit I/O group represents 8 bits (1 byte). Each bit represents a bit I/O port. The least significant bit (LSB) represents port 0. When a bit is set, the related bit I/O port is configured for the mode specified.

Mode Specifies the mode for which the related bit I/O ports have to be initialized:

- DSRPCU_ADDIO_BIO_INPUT_MODE: Associated ports are configured as inports.
- DSRPCU_ADDIO_BIO_OUTPUT_MODE: Associated ports are configured as output ports.

InitData Specifies the initial output pin state (only 0 and 1 is allowed). For inports, this parameter is ignored.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7700	Error	ADDIO: memory allocation error.	The memory allocation for internal data storage failed.
0x7720	Error	ADDIO Bit-IO Init: illegal group number specified.	The specified value is invalid. Use one of the predefined symbols to specify the GroupNo parameter.
0x7721	Error	ADDIO Bit-IO Init: illegal direction mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the Mode parameter.
0x7722	Error	ADDIO Bit-IO Init: multiple initialization of Bit-IO port.	You can initialize a Bit-IO port only once.
0x3702	Info	ADDIO Bit-IO Init: group number %d initialized.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_bio_req_read....76
dsrpcu_bio_request....75
dsrpcu_bio_write....74

dsrpcu_bio_write

Syntax

```
Int16 dsrpcu_bio_write (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 Data)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To perform write access to the bit I/O ports.

Description

The bit I/O ports are specified via the ParamSetPtr parameter.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in

ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Data Represents the bit values to be written for each bit I/O port. Each single bit represents one bit I/O port. The least significant bit (LSB) represents port 0.

Note

Only bits initialized for output mode in the relevant parameter set are used, refer to **Mode** on page 73.
You must maintain that the buffer size is sufficient.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values could not be transferred to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_bio_init2 72
dsrpcu_bio_req_read 76
dsrpcu_bio_request 75

dsrpcu_bio_request

Syntax

```
Int16 dsrpcu_bio_request (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To request the most recent pin state.

Description

This function performs a request for the most recent pin state of a bit I/O channel. The requested data can be read by using the **`dsrpcu_bio_req_read`** function. The bit I/O ports are specified via the ParamSetPtr parameter.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_bio_init2 72
dsrpcu_bio_req_read 76
dsrpcu_bio_write 74

dsrpcu_bio_req_read

Syntax

```
Int16 dsrpcu_bio_req_read (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   UInt8* Data,
   UInt8* Status)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To read the most recent pin state in request-read measurement mode.

Description

This function performs read access to the most recent pin state of a bit I/O channel. The bit I/O ports are specified via the ParamSetPtr parameter.

Note

This function must be called in combination with dsrpcu_bio_request().

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer data from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_ADDIO_BIT_READ_MODE_CURRENT: dsrpcu_bio_req_read() just returns the current available data.
- DSRPCU_ADDIO_BIT_READ_MODE_NEW: The function polls until requested data is available. dsrpcu_bio_req_read() returns the requested data.

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

The bit I/O channels are read by the slave, and the data is written into the DPMEM.

Data Pointer to an address where the read values are stored. Each single bit represents one bit I/O port. The least significant bit (LSB) represents port 0.

Note

Bits that have not been initialized for input mode in the relevant parameter set are set to 0, refer to **Mode** on page 73.
You must maintain that the buffer size is sufficient.

Status Address where the current status of the pin state is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

DSRPCU_ADDIO_ERR_BIO_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_bio_init2 72
dsrpcu_bio_request 75
dsrpcu_bio_write 74

Bit I/O via TPU

Interrupts

Processing digital signals via a TPU unit allows interrupts to be transferred to the master. This is not possible when digital signals are processed via the I/O PLD (Programmable Logic Device).

Where to go from here

Information in this section

dsrpcu_tpu_digout_init

Syntax

```
void dsrpcu_tpu_digout_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 Priority,
   UInt16 PinState)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize a TPU channel as digital outport.

Description

If more than one digital output channel is addressed, this function must be called several times with the right channel and unit numbers. Each function call then requires its own parameter set.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

PinState The initial output pin state (only 0 and 1 is allowed).

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7FC0	Error	TPU DigOut Init: wrong protocol version. Update application firmware.	The use of this functions requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7FC1	Error	TPU DigOut Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7FC2	Error	TPU DigOut Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.

ID	Type	Message[1)]	Description
0x7FC3	Error	TPU DigOut Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7FC4	Error	TPU DigOut Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7FC5	Error	TPU DigOut Init: illegal pin state specified.	The specified value is invalid. Use one of the predefined symbols to specify the PinState parameter.
0x7F06	Error	TPU: memory allocation error.	An error occurs during memory allocation.
0x3F14	Info	TPU Init: DigOUT Init: channel X on TPU X is initialized as digital out.	The TPU DigOUT channel was initialized successfully. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_digout_write.....81

dsrpcu_tpu_digout_write

Syntax

```
Int16 dsrpcu_tpu_digout_write(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt16* PinState)
```

Include file

```
dsrpcutpu.h
```

Purpose

To perform write access to a TPU channel.

Description

This function updates the pin state of a TPU channel, which has been initialized as digital output before.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

PinState The output pin state which has to be sent to the slave (only 0 and 1 is allowed).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The next free parameter set is blocked (only in FIFO mode).

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_tpu_digout_init..... 79

dsrpcu_tpu_digin_init2

Syntax

```
void dsrpcu_tpu_digin_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 Priority,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize TPU channels as digital inports.

Description

If more than one digital input channel is addressed, this function must be called several times with the right channel and unit numbers. Each function call then requires its own parameter set.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

MeasureMode Specifies the measurement mode:

- DSRPCU_TPU_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_tpu_digin_read.
- DSRPCU_TPU_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. For reading the results, use dsrpcu_tpu_digin_request and dsrpcu_tpu_digin_req_read.

IntEnable Defines whether an interrupt is to be triggered by the slave when an edge of the input signal occurs. The edge polarity is defined by the EdgePol parameter.

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

EdgePol Matching edge of the input signal which causes an interrupt on the master:

- DSRPCU_TPU_DIGIN_RISING
- DSRPCU_TPU_DIGIN_FALLING
- DSRPCU_TPU_DIGIN_EITHER

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F3F	Error	TPU DigIn Init: wrong protocol version. Update application firmware.	The use of this functions requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7F40	Error	TPU DigIn Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F41	Error	TPU DigIn Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F42	Error	TPU DigIn Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F43	Error	TPU DigIn Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F44	Error	TPU DigIn Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter..
0x7F45	Error	TPU DigIn Init: illegal edge polarity parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7F46	Error	TPU DigIn Init: illegal measurement mode parameter specified	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x3F06	Info	TPU DigIn Init: channel X on TPU X is initialized as digital in.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_tpu_digin_read....85
dsrpcu_tpu_digin_req_read....87
dsrpcu_tpu_digin_request....86

dsrpcu_tpu_digin_read

Syntax

```
Int16 dsrpcu_tpu_digin_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt16* PinState,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent pin state in continuous measurement mode.

Description

This function performs a read access to the most recent and the latest 15 previous pin states of a TPU channel. The bit I/O ports are specified via the ParamSetPtr parameter.

Note

This function must be called in continuous measurement mode. Thus, it must not be called in combination with dsrpcu_tpu_digin_request().

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The

parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

PinState Address where the pin state is stored. The most recent pin state is stored in the MSB (Most Significant Bit). The 15 previous pin states are stored in the subsequent bits

Status Address where the current status of the pin state is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

dsrpcu_tpu_digin_init2 82
dsrpcu_tpu_digin_req_read 87
dsrpcu_tpu_digin_request 86

dsrpcu_tpu_digin_request

Syntax

```
Int16 dsrpcu_tpu_digin_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To request the most recent pin state in request-read measurement mode.

Description

This function performs a request of the most recent and the latest 15 previous pin states of a TPU channel. The requested data will be provided by the slave. The data can be read via dsrpcu_tpu_digin_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_digin_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_tpu_digin_init2 82
dsrpcu_tpu_digin_read 85
dsrpcu_tpu_digin_req_read 87

dsrpcu_tpu_digin_req_read

Syntax

```
Int16 dsrpcu_tpu_digin_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   UInt16* PinState,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent pin state in request-read measurement mode.

Description

This function performs a read access to the most recent and the latest 15 previous pin states of a TPU channel. The bit I/O ports are specified via the ParamSetPtr parameter.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_digin_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the pin status from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_TPU_READ_MODE_CURRENT: Read current available data
- DSRPCU_TPU_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

PinState Address where the pin state is stored. The most recent pin state is stored in the MSB (Most Significant Bit). The 15 previous pin states are stored in the subsequent bits

Status Address where the current status of the pin state is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_TPU_ERR_DIGIN_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

Timing I/O

Objective

The RapidPro Control Unit RTLib provides functions that you can use for generating and measuring signals.

PWM signals

In the following descriptions, the expression *PWM signals* means pulse width modulated square-wave signals.

Where to go from here

Information in this section

PWM Signal Generation (PWM)

Objective

To generate single, edge-aligned, and center-aligned PWM signals.

MIOS and TPU

PWM signal generation is possible via MIOS and TPU. For the characteristics of these two methods, refer to Comparing TPU and MIOS (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Where to go from here

Information in this section

Information in other sections

Introduction to Rapid Control Prototyping with the RapidPro System (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_mios_pwm_init

Syntax

```
void dsrpcu_mios_pwm_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Channel,
   UInt16 UpdateMode,
   UInt16 ChPrescaler,
   UInt16 Polarity,
   Float64 Period,
   Float64 Duty)
```

Include file

```
dsrpcumios.h
```

Purpose

To initialize PWM signal generation on a MIOS channel.

Description

If more than one PWM signal is to be generated, this function must be called several times, with different channel numbers. Each function call requires its own parameter set.

The possible period range (T_{min} … T_{max}) follows the following formula:

T_{min}	2 * GP * (256 - CP) / CF
T_{max}	65535 * GP * (256 - CP) / CF
GP	Global prescaler: [2 … 16]. Adjusted via dsrpcu_mios_init.
CP	Channel prescaler: [0 … 255]. Adjusted via the ChPrescaler parameter.
CF	CPU clock frequency: 56 MHz

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the frequency and the duty cycle from the master to the slave processor. The parameter set is assigned to a specific channel and initialized when the function is being executed.

Channel MIOS channel number [1 … 12]. The function uses one channel.

UpdateMode Defines whether only the duty cycle (fixed period) or the duty cycle and the period (variable period) can be updated via the update functions:

- DSRPCU_MIOS_PWM_FIX_PERIOD
- DSRPCU_MIOS_PWM_VAR_PERIOD

ChPrescaler Prescaler of the MIOS channel [0 … 255].

Polarity Defines the polarity of the PWM signal:

- DSRPCU_MIOS_PWM_HIGH_ACTIVE
- DSRPCU_MIOS_PWM_LOW_ACTIVE

Note

The polarity depends on the configuration of the SC modules used and the configuration settings in ConfigurationDesk for RapidPro.

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$.

Duty Duty cycle of the PWM [0.0 … 1.0]

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7A0F	Error	MIOS: memory allocation error.	The memory allocation for internal data storage failed.
0x7A10	Error	MIOS PWM Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7A11	Error	MIOS PWM Init: illegal channel prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the ChPrescaler parameter.
0x7A12	Error	MIOS PWM Init: illegal update mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the UpdateMode parameter.

ID	Type	Message[1)]	Description
0x7A13	Error	MIOS PWM Init: illegal value for polarity specified.	The specified value is invalid. Use one of the predefined symbols to specify the Polarity parameter.
0x7A14	Error	MIOS PWM Init: illegal value for the period specified.	The specified value is invalid. The specified value is outside the range permitted for the Period parameter.
0x7A15	Error	MIOS PWM Init: illegal value for the duty cycle specified.	The specified value is invalid. Use one of the predefined symbols to specify the Duty parameter.
0x3A01	Info	MIOS PWM Init: channel X is initialized as PWM.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_mios_pwm_fp_update....97
dsrpcu_mios_pwm_start....95
dsrpcu_mios_pwm_vp_update....96

dsrpcu_mios_pwm_start

Syntax

```
void dsrpcu_mios_pwm_start(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
dsrpcumios.h
```

Purpose

To start PWM signal generation on all initialized MIOS channels.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

None

Related topics

References

dsrpcu_mios_pwm_vp_update

Syntax

```
Int16 dsrpcu_mios_pwm_vp_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Period,
   Float64 Duty)
```

Include file

```
dsrpcumios.h
```

Purpose

To update the period and duty cycle of the PWM channel specified by the ParamSetPtr parameter during run-time.

Note

This function can be used only if the UpdateMode parameter of the dsrpcu_mios_pwm_init function has been set to DSRPCU_MIOS_PWM_VAR_PERIOD.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle from the master to the slave

processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$. For information on the period range, refer to dsrpcu_mios_pwm_init on page 93.

Duty Duty cycle of the PWM [0.0 … 1.0]

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_mios_pwm_fp_update

Syntax

```
Int16 dsrpcu_mios_pwm_fp_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Duty)
```

Include file

```
dsrpcumios.h
```

Purpose

To update the duty cycle of the PWM channel specified by the ParamSetPtr parameter during run-time.

Note

This function can be used only if the UpdateMode parameter of the dsrpcu_mios_pwm_init function has been set to DSRPCU_MIOS_PWM_VAR_PERIOD.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the duty cycle from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Duty Duty cycle of the PWM [0.0 … 1.0]

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_mios_pwm_init....93
dsrpcu_mios_pwm_start....95
dsrpcu_mios_pwm_vp_update....96

dsrpcu_tpu_pwm_init

Syntax

```
void dsrpcu_tpu_pwm_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 Polarity,
   Float64 Period,
   Float64 Duty,
   UInt16 IntRate)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize PWM signal generation on a defined TPU and a defined channel.

Description

The values for the period and the duty cycle at initialization are defined. If more than one PWM is to be generated, this function must be called several times, with different channel and/or unit numbers. Each function call requires its own parameter set.

Range values

The possible period range [T_{min} … T_{max}] and the TPU timer resolution (TR) follow the following formula:

T_{min}	= 0x001C * TR
T_{max}	= 0x7FFF * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period ranges [T_{min} … T_{max}] and TPU timer resolution are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	1.00	1.17	855	1000	35.7
4	–	2.00	2.34	427	500	71.4

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
8	8	4.0	4.68	214	250	143
14	–	7.0	8.19	122	142.9	250
–	16	8.0	9.36	107	125.0	286
–	24	12.0	14.0	71	83.3	429
28	–	14.0	16.4	61	71.4	500
–	32	16.0	18.7	53	62.5	571
42	–	21.0	24.6	41	47.6	750
56	56	28.0	32.8	31	35.7	1000
–	64	32.0	37.4	27	31.3	1140
84	–	42	49.2	20.3	23.81	1500
112	–	56	65.5	15.3	17.86	2000
–	120	60	70.2	14.2	16.67	2140
168	–	84	98.3	10.2	11.90	3000
224	–	112	131	7.6	8.93	4000
336	–	168	197	5.1	5.95	6000
448	–	224	262	3.8	4.46	8000

Note

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle update values from the master to the slave processor during run-time. The

parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:
- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

PrescalerSel Prescaler selection:
- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:
- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Polarity Defines whether a high or a low active PWM signal is to be generated:
- DSRPCU_TPU_PWM_HIGH_ACTIVE
- DSRPCU_TPU_PWM_LOW_ACTIVE

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$.

Duty Duty cycle of the PWM [0.0 … 1.0]

IntRate Number of PWM periods after which a TPU interrupt is generated [1 ... 256] and sent to the master. If no interrupt is required, IntRate must be set to DSRPCU_TPU_INT_DISABLE.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F1C	Error	TPU PWM Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F1D	Error	TPU PWM Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F1E	Error	TPU PWM Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F1F	Error	TPU PWM Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.

ID	Type	Message[1)]	Description
0x7F20	Error	TPU PWM Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F21	Error	TPU PWM Init: illegal value for polarity specified.	The specified value is invalid. Use one of the predefined symbols to specify the Polarity parameter.
0x7F22	Error	TPU PWM Init: illegal value for the period specified.	The specified value is invalid. The specified value is outside the range permitted for the Period parameter.
0x7F23	Error	TPU PWM Init: illegal value for the duty cycle specified.	The specified value is invalid. Use one of the predefined symbols to specify the Duty parameter.
0x7F24	Error	TPU PWM Init: illegal value for the interrupt rate specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntRate parameter.
0x3F02	Info	TPU PWM Init: channel %d on TPU %d is initialized as PWM.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_pwm_update..... 102
RPCU_TIMER_SETUP_TPU_BLx (RapidPro System – I/O Subsystem MPC565 RTI Reference 🕮)

dsrpcu_tpu_pwm_update

Syntax

```
Int16 dsrpcu_tpu_pwm_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Period,
   Float64 Duty)
```

Include file

```
dsrpcutpu.h
```

Purpose

To update the period and duty cycle of the PWM channel specified by the ParamSetPtr parameter during run-time.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle update values from the master to the slave processor during run-time. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$.

Duty Duty cycle of the PWM [0.0 … 1.0]

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_pwm_init.....99

dsrpcu_tpu_mcpwm_ea_init

Syntax

```
void dsrpcu_tpu_mcpwm_ea_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   Float64 Period,
   Float64 Duty1,
   Float64 Duty2,
   Float64 Duty3,
   UInt16 IntMode,
   UInt16 IntRate,
   Float64 IntPosition)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize the generation of edge-aligned multi-channel PWM signals on a defined TPU.

Description

This function is used to initialize the generation of three edge-aligned PWM signals. Additionally, interrupts and an interrupt trigger signal can be generated. The generation of edge-aligned multi-channel PWM signals is possible on each of the three TPU units A, B, and C separately. However, only one `dsrpcu_tpu_mcpwm_ea_init` function can be executed on one TPU unit. If more than one PWM signal is to be generated, this function must be called several times, with a different unit number. Each function call requires its own parameter set.

Range values

The possible period range (T_{min} … T_{max}) follows the following formula:

T_{min}	= 0x008C * TR
T_{max}	= 0x4000 * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period ranges [T_{min} … T_{max}] are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	5.00	0.59	1709	200	35.7
4	–	10.0	1.17	854	100	71.4
8	8	20.0	2.34	427	50	143
14	–	35.0	4.10	244	28.6	250
–	16	40.0	4.68	214	25.0	286
–	24	60.0	7.02	142	16.7	429
28	–	70.0	8.19	122	14.3	500
–	32	80.0	9.36	107	12.5	571
42	–	105	12.29	81.4	9.5	750
56	56	140	16.38	61.0	7.1	1000
–	64	160	18.72	53.4	6.3	1140

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
84	–	210	24.6	40.7	4.76	1500
112	–	280	32.8	30.5	3.57	2000
–	120	300	35.1	28.5	3.33	2140
168	–	420	49.2	20.3	2.38	3000
224	–	560	65.5	15.3	1.79	4000
336	–	840	98.3	10.2	1.19	6000
448	–	1120	131.1	7.63	0.89	8000

Note

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

Channel usage

Address the TPU channel TPU_CH1, which is used as the master channel. Three successive TPU channels are automatically addressed as PWM output channels. Whether an additional TPU channel is used for interrupt generation depends on the IntMode on page 107 parameter. The original TPU output signals are processed by additional logic inside the I/O PLD in order to form the output signals for the I/O bus. The following table gives an overview of the resulting I/O bus output signals:

I/O Bus Output Signal	IntMode = DSRPCU_TPU_MCPWM_xxx	
	xxx = VAR_POS_INT_ENABLE	***xxx = FIX_POS_INT_ENABLE*** ***xxx = INT_DISABLE***
TPU_CH1	Master channel	Master channel
TPU_CH2	PWM output channel (a+)	PWM output channel (a+)
TPU_CH3	PWM output channel (b+)	PWM output channel (b+)
TPU_CH4	PWM output channel (c+)	PWM output channel (c+)
TPU_CH5	Used for the additionally generated PWM signal (interrupt trigger signal)	Not used

Interrupt generation

The PWM periods for all channels start with the rising edge. The interrupt position is as follows:

- If the fixed position interrupt is selected, that is, `IntMode = DSRPCU_TPU_MCPWM_FIX_POS_INT_ENABLE`, the interrupt is triggered at the rising edge of the PWM signals.

- If a variable position interrupt is selected, that is, `IntMode = DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE`, the `IntPosition` parameter specifies the delay between the rising edge of the PWM signals and the point in time when the interrupt is triggered.

Note

For the use of variable interrupts, an additional PWM signal is generated which is specified as follows (for details, refer to Basics of Edge-Aligned Multi-Channel PWM Signal Generation (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)):

- Rising edge at the same position as the variable interrupt
  By specifying the number of PWM periods after which a variable interrupt is periodically generated (see IntRate parameter), you also specify the number of PWM periods after which the rising edge of the additional PWM signal is periodically generated.
- Falling edge at the middle of the period of the edge-aligned multi-channel PWM signal

Due to the master-slave communication it might happen that an interrupt reaches the master processor a few micro seconds later than specified.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle update values from the master to the slave processor during run-time. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Number of the first TPU channel used. Must always be set to TPU_CH1. The function uses several channels, refer to Channel usage on page 105.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$.

Duty1 Duty cycle of the 1st PWM output channel [0.0 … 1.0].

Duty2 Duty cycle of the 2nd PWM output channel [0.0 … 1.0].

Duty3 Duty cycle of the 3rd PWM output channel [0.0 … 1.0].

IntMode Interrupt type triggered by the MCPWM signal:

- DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE
- DSRPCU_TPU_MCPWM_FIX_POS_INT_ENABLE
- DSRPCU_TPU_MCPWM_INT_DISABLE

IntRate Number of PWM periods after which a TPU interrupt is generated [1 ... 256].

IntPosition Delay (in seconds) between the rising edge of the PWM signals and the point in time when the interrupt is triggered. Range: [0 … Period]. This parameter is evaluated only if IntMode is set to DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE. If not, set IntPosition to zero.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F5C	Error	TPU MCPWM EA Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F5D	Error	TPU MCPWM EA Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F5E	Error	TPU MCPWM EA Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F5F	Error	TPU MCPWM EA Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.

ID	Type	Message[1)]	Description
0x7F60	Error	TPU MCPWM EA Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F61	Error	TPU MCPWM EA Init: illegal value for the period specified	The specified value is invalid. The specified value is outside the range permitted for the Period parameter.
0x7F62	Error	TPU MCPWM EA Init: illegal value for the duty cycle specified.	The specified value is invalid. Use one of the predefined symbols to specify the Duty1, Duty2, or Duty3 parameter.
0x7F63	Error	TPU MCPWM EA Init: illegal value for the interrupt rate specified.	The specified value is invalid. The specified value is outside the range permitted for the IntRate parameter.
0x7F64	Error	TPU MCPWM EA Init: illegal value for the interrupt mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntMode parameter.
0x7F65	Error	TPU MCPWM EA Init: illegal value for the interrupt position time specified.	The specified value is invalid. The specified value is outside the range permitted for the IntPosition parameter.
0x7F66	Error	TPU MCPWM EA Init: illegal value for the channel number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Channel parameter.
0x3F0A	Info	TPU MCPWM EA Init: channel %d - %d on TPU %d are initialized as MCPWM EA.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

Basics

Edge-Aligned Multi-Channel PWM Signal Generation (MCPWM_EA) (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_tpu_mcpwm_ca_init..... 110
dsrpcu_tpu_mcpwm_ca_update..... 116
dsrpcu_tpu_mcpwm_ea_update..... 108

dsrpcu_tpu_mcpwm_ea_update

Syntax

```
Int16 dsrpcu_tpu_mcpwm_ea_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Duty1,
   Float64 Duty2,
   Float64 Duty3)
```

Include file	`dsrpcutpu.h`
Purpose	To update the duty cycles of edge-aligned PWM signals specified by the ParamSetPtr parameter during run-time.
Description	The duty cycles of the three PWM signals can be changed separately at run time, and are consecutively updated at the beginning of a new PWM period.
I/O mapping	The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **ParamSetPtr** Start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle update values from the master to the slave processor during run-time. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`). **Duty1** Update value for the duty cycle of the 1st PWM output channel [0.0 … 1.0]. **Duty2** Update value for the duty cycle of the 2nd PWM output channel [0.0 … 1.0]. **Duty3** Update value for the duty cycle of the 3rd PWM output channel [0.0 … 1.0].
Return value	**DSRPCU_NO_ERROR** No error occurred. **DSMSC_PARAM_SET_ACCESS_ERROR** Master-slave communication error. The update values have not been sent to the slave. **DSMSC_CMD_BUFFER_OVERFLOW** Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_mcpwm_ca_init..... 110
dsrpcu_tpu_mcpwm_ca_update..... 116
dsrpcu_tpu_mcpwm_ea_init..... 103

dsrpcu_tpu_mcpwm_ca_init

Syntax

```
void dsrpcu_tpu_mcpwm_ca_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 PWMMode,
   Float64 Period,
   Float64 Duty1,
   Float64 Duty2,
   Float64 Duty3,
   Float64 DeadTime,
   UInt16 IntMode,
   UInt16 IntRate,
   Float64 IntPosition)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize the generation of center-aligned multi-channel PWM signals on a defined TPU.

Description

This function is used to initialize the generation of three or six center-aligned PWM signals. Additionally, interrupts and an interrupt trigger signal can be generated. The number of output signals depends on the PWMMode and IntMode parameters. The generation of center-aligned multi-channel PWM signals is possible on each of the three TPU units A, B, and C separately. However, only one `dsrpcu_tpu_mcpwm_ea_init` function can be executed on one TPU unit. If more than one PWM signal is to be generated, this function must be called several times, with a different unit number. Each function call requires its own parameter set.

Range values

The possible period ranges [T_{min} … T_{max}] and dead time values follow the following formula:

T_{min}	= 0x0118 * TR
T_{max}	= 0x4000 * TR
Dead Time	= 0x00FF * TR / 2
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period ranges [T_{min} … T_{max}] and maximum dead time values are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Dead Time_max [µs]	Resolution [ns]
2	–	10.0	0.59	1709	100	4.55	35.7
4	–	20.0	1.17	854	50	9.11	71.4
8	8	40.0	2.34	427	25	18.2	143
14	–	70.0	4.10	244	14.3	31.9	250
–	16	80.0	4.68	214	12.5	36.4	286
–	24	120	7.02	142	8.3	54.6	429
28	–	140	8.19	122	7.1	63.8	500
–	32	160	9.36	107	6.3	72.9	571
42	–	210	12.3	81.4	4.8	95.6	750
56	56	280	16.4	61.0	3.6	128	1000
–	64	320	18.7	53.4	3.1	146	1140
84	–	420	24.6	40.7	2.38	191	1500
112	–	560	32.8	30.5	1.79	255	2000
–	120	600	35.1	28.5	1.67	273	2140
168	–	840	49.2	20.3	1.19	383	3000
224	–	1120	65.5	15.3	0.89	510	4000
336	–	1680	98.3	10.2	0.60	765	6000
448	–	2240	131	7.63	0.45	1020	8000

Note

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

Channel usage

Address the TPU channel TPU_CH1, which is used as the master channel. Successive TPU channels are automatically addressed as PWM output channels. Whether an additional TPU channel is used for interrupt generation depends on the IntMode parameter. The original TPU output signals are processed by additional logic inside the I/O PLD in order to form the output signals for the I/O bus. The following tables give an overview of the resulting I/O bus output signals:

- PWMMode = DSRPCU_TPU_MCPWM_PWM3_MODE:

I/O Bus Output Signal	**IntMode = DSRPCU_TPU_MCPWM_xxx**	
	xxx = VAR_POS_INT_ENABLE	***xxx = FIX_POS_INT_ENABLE*** ***xxx = INT_DISABLE***
TPU_CH1	Master channel, no output signal	Master channel, no output signal
TPU_CH2	PWM a+	PWM a+
TPU_CH3	Reserved	Reserved
TPU_CH4	PWM b+	PWM b+
TPU_CH5	Reserved	Reserved
TPU_CH6	PWM c+	PWM c+
TPU_CH7	Reserved	Reserved
TPU_CH8	Used for the additionally generated PWM signal (interrupt trigger signal)	Allocated, but not used for output signal
TPU_CH9	Allocated, but not used for output signal	–

- PWMMode = DSRPCU_TPU_MCPWM_PWM6_MODE. All non-inverted signals include the dead time:

I/O Bus Output Signal	**IntMode = DSRPCU_TPU_MCPWM_xxx**	
	xxx = VAR_POS_INT_ENABLE	***xxx = FIX_POS_INT_ENABLE*** ***xxx = INT_DISABLE***
TPU_CH1	Master channel, no output signal	Master channel, no output signal
TPU_CH2	PWM a-	PWM a-
TPU_CH3	Reserved	Reserved
TPU_CH4	PWM a+ Includes the dead time	PWM a+ Includes the dead time
TPU_CH5	Reserved	Reserved
TPU_CH6	PWM b-	PWM b-
TPU_CH7	Reserved	Reserved
TPU_CH8	PWM b+ Includes the dead time	PWM b+ Includes the dead time

I/O Bus Output Signal	IntMode = DSRPCU_TPU_MCPWM_xxx	
	xxx = VAR_POS_INT_ENABLE	***xxx = FIX_POS_INT_ENABLE*** ***xxx = INT_DISABLE***
TPU_CH9	Reserved	Reserved
TPU_CH10	PWM c-	PWM c-
TPU_CH11	Reserved	Reserved
TPU_CH12	PWM c+ Includes the dead time	PWM c+ Includes the dead time
TPU_CH13	Reserved	Reserved
TPU_CH14	Used for the additionally generated PWM signal (interrupt trigger signal)	Allocated, but not used for output signal
TPU_CH15	Allocated, but not used for output signal	–

The PWM periods for all channels start with the center position of the low times. The interrupt position is as follows:

- If the fixed position interrupt is selected, that is, `IntMode = DSRPCU_TPU_MCPWM_FIX_POS_INT_ENABLE`, the interrupt is triggered at the center position of the high times.
- If a variable position interrupt is selected, that is, `IntMode = DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE`, the `IntPosition` parameter specifies the delay between the center position of the low times and the point in time when the interrupt is triggered.

Note

For the use of variable interrupts, an additional PWM signal is generated which is specified as follows (for details, refer to Basics of Center-Aligned Multi-Channel PWM Signal Generation (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)):

- Rising edge at the same position as the variable interrupt
  By specifying the number of PWM periods after which a variable interrupt is periodically generated (see IntRate parameter), you also specify the number of PWM periods after which the rising edge of the additional PWM signal is periodically generated.
- Falling edge at the middle of the period of the edge-aligned multi-channel PWM signal

Due to the master-slave communication it might happen that an interrupt reaches the master processor a few micro seconds later than specified.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the duty cycle update values from the master to the slave processor during run-time. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel This parameter is for future use. Set it to TPU_CH1 in order to avoid an error message.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

PWMMode This parameter specifies whether all PWM signals are available as inverted signals, too:

- DSRPCU_TPU_MCPWM_PWM3_MODE
- DSRPCU_TPU_MCPWM_PWM6_MODE

Period Period T_p of the PWM (in seconds). T_p must fit the related period range: $T_{min} < T_p < T_{max}$.

Duty1 Duty cycle of the 1st PWM output channel [0.0 … 1.0].

Duty2 Duty cycle of the 2nd PWM output channel [0.0 … 1.0].

Duty3 Duty cycle of the 3rd PWM output channel [0.0 … 1.0].

DeadTime Dead time of the non-inverted PWM signals (PWMMode = DSRPCU_TPU_MCPWM_PWM6_MODE). The dead time range depends on the prescaler settings, see tables above. If PWMMode is set to DSRPCU_TPU_MCPWM_PWM3_MODE, set DeadTime to zero.

IntMode Interrupt type triggered by the MCPWM signal:

- DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE
- DSRPCU_TPU_MCPWM_FIX_POS_INT_ENABLE
- DSRPCU_TPU_MCPWM_INT_DISABLE

IntRate Number of PWM periods after which a TPU interrupt is generated [1 ... 256].

IntPosition Delay (in seconds) between the beginning of a PWM period and the point in time when the interrupt is triggered. Range: [0 … Period]. A PWM period begins at the center of a PWM low time. This parameter is evaluated only if IntMode is set to DSRPCU_TPU_MCPWM_VAR_POS_INT_ENABLE. If not, set IntPosition to zero.

Return value None

Messages The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F68	Error	TPU MCPWM CA Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F69	Error	TPU MCPWM CA Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F6A	Error	TPU MCPWM CA Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F6B	Error	TPU MCPWM CA Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.
0x7F6C	Error	TPU MCPWM CA Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F6D	Error	TPU MCPWM CA Init: illegal value for the period specified.	The specified value is invalid. The specified value is outside the range permitted for the Period parameter.
0x7F6E	Error	TPU MCPWM CA Init: illegal value for the duty cycle specified.	The specified value is invalid. Use one of the predefined symbols to specify the Duty1, Duty2, or Duty3 parameter.
0x7F6F	Error	TPU MCPWM CA Init: illegal value for the interrupt rate specified.	The specified value is invalid. The specified value is outside the range permitted for the IntRate parameter.
0x7F70	Error	TPU MCPWM CA Init: illegal value for the interrupt mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntMode parameter.
0x7F71	Error	TPU MCPWM CA Init: illegal value for the interrupt position time specified.	The specified value is invalid. The specified value is outside the range permitted for the IntPosition parameter.
0x7F72	Error	TPU MCPWM CA Init: illegal value for the PWM mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the PWMMode parameter.
0x7F73	Error	TPU MCPWM CA Init: illegal value for the dead time specified.	The specified value is invalid. Use one of the predefined symbols to specify the DeadTime parameter.

ID	Type	Message[1)]	Description
0x3F0B	Info	TPU MCPWM CA Init: channel %d - %d on TPU %d are initialized as MCPWM CA.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

Basics

Center-Aligned Multi-Channel PWM Signal Generation (MCPWM_CA) (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_tpu_mcpwm_ca_update..... 116
dsrpcu_tpu_mcpwm_ea_init..... 103
dsrpcu_tpu_mcpwm_ea_update..... 108

dsrpcu_tpu_mcpwm_ca_update

Syntax

```
Int16 dsrpcu_tpu_mcpwm_ca_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Duty1,
   Float64 Duty2,
   Float64 Duty3)
```

Include file

```
dsrpcutpu.h
```

Purpose

To update the duty cycle of the center-aligned PWM signals specified by the ParamSetPtr parameter during run-time.

Description

The duty cycles of the signals can be changed separately at run time, and are simultaneously updated at the beginning of a new PWM period.

Note

Simultaneous updating of all three duty cycles is possible down to a period of 50 µs (PWM6_MODE) and 45 µs (PWM3_MODE). If faster periods are desired, simultaneous updating cannot be guaranteed.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the period and the duty cycle update values from the master to the slave processor during run-time. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Duty1 Initial duty cycle of the 1st PWM output channel [0 … 1]

Duty2 Initial duty cycle of the 2nd PWM output channel [0 … 1]

Duty3 Initial duty cycle of the 3rd PWM output channel [0 … 1]

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_mcpwm_ca_init..... 110
dsrpcu_tpu_mcpwm_ea_init..... 103
dsrpcu_tpu_mcpwm_ea_update..... 108

PWM Signal Measurement (PWM2D)

Objective

To measure the frequency and duty cycle of a single PWM signal.

Where to go from here

Information in this section

dsrpcu_tpu_pwm2d_init2

Syntax

```
void dsrpcu_tpu_pwm2d_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize the measurement of PWM signals on a defined TPU and defined channels.

Description

This function measures the frequency and the duty cycle of a PWM signal. If more than one PWM signal is to be generated, this function must be called several times, with different channel and/or unit numbers. Each function call requires its own parameter set.

The maximum measured frequency depends on the number of used PWM2D channels and the use of other components of the DIO. The Fmax values of 100 kHz can only be reached when no other functions are running on the TPU.

Range values

The possible period range [T_{min} … T_{max}] follows the following formula:

T_{min}	= 0x0028 * TR
T_{max}	= 0x7FFF * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period range numbers are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	1.43	1.17	855	700	35.7
4	–	2.86	2.34	427	350	71.4
8	8	5.7	4.68	214	175	143
14	–	10.0	8.19	122	100.0	250
–	16	11.4	9.36	107	87.5	286
–	24	17.1	14.04	71	58.3	429
28	–	20.0	16.38	61	50.0	500
–	32	22.9	18.72	53	43.8	571
42	–	30.0	24.6	41	33.3	750
56	56	40.0	32.8	31	25.0	1000
–	64	45.7	37.4	27	21.9	1140
84	–	60	49.2	20.3	16.67	1500
112	–	80	65.5	15.3	12.50	2000
–	120	86	70.2	14.2	11.67	2140
168	–	120	98.3	10.2	8.33	3000
224	–	160	131.1	7.6	6.25	4000

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
336	–	240	196.6	5.1	4.17	6000
448	–	320	262	3.8	3.13	8000

Note

The ranges are affected by the TPU workload. If the TPU executes this function on more than one channel, or executes additional functions, the maximum value may not be reached.

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the frequency and the duty cycle from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

MeasureMode Specifies the measurement mode:

- DSRPCU_TPU_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_tpu_pwm2d_read.
- DSRPCU_TPU_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. For reading the results, use dsrpcu_tpu_pwm2d_request and dsrpcu_tpu_pwm2d_req_read.

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

EdgePol Matching edge of the pulse for signal detection:

- DSRPCU_TPU_PWM2D_RISING_EDGE
- DSRPCU_TPU_PWM2D_FALLING_EDGE

Return value None

Messages The following messages are defined:

ID	Type	Message[1)]	Description
0x7F47	Error	TPU PWM2D Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7F48	Error	TPU PWM2D Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F49	Error	TPU PWM2D Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F4A	Error	TPU PWM2D Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F4B	Error	TPU PWM2D Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.
0x7F4C	Error	TPU PWM2D Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F4D	Error	TPU PWM2D Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7F4E	Error	TPU PWM2D Init: illegal edge polarity parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7F4F	Error	TPU PWM2D Init: illegal measurement mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.

ID	Type	Message[1)]	Description
0x3F07	Info	TPU PWM2D Init: channel X on TPU X is initialized as PWM measurement.	The TPU channel was initialized successfully for PWM2D measurement. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_tpu_pwm2d_read..... 122
dsrpcu_tpu_pwm2d_req_read..... 125
dsrpcu_tpu_pwm2d_request..... 124

dsrpcu_tpu_pwm2d_read

Syntax

```
Int16 dsrpcu_tpu_pwm2d_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* Frequency,
   Float64* Duty,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the measured duty cycle and frequency values of a specific TPU channel in continuous measurement mode.

Description

If the duty cycle is 0.0 or 1.0, the Frequency parameter is always set to zero and an error code is returned. If the measured frequency is below the lower range limit, the Frequency parameter contains a zero (for the lower range limit, refer to Range values on page 119). In this case the duty cycle is undefined because it is not clear which signal polarity the input had, and the function returns an error code. If the measured frequency is too high, the function returns unpredictable results.

Note

This function must be called in continuous measurement mode. Thus, it must not be called in combination with dsrpcu_tpu_pwm2d_request().

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the frequency and the duty cycle from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Frequency Address where the measured frequency is stored.

Duty Pointer to an address where the measured duty cycle is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The input signal is always high or low or the frequency exceeds the range.

Related topics

References

dsrpcu_tpu_pwm2d_init2 118
dsrpcu_tpu_pwm2d_req_read 125
dsrpcu_tpu_pwm2d_request 124

dsrpcu_tpu_pwm2d_request

Syntax

```
Int16 dsrpcu_tpu_pwm2d_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To request the measured duty cycle and frequency values of a specific TPU channel in request-read measurement mode.

Description

This function performs a request of pulse width measurement data. The requested data will be provided by the slave. The data can be read via dsrpcu_tpu_pwm2d_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_pwm2d_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the frequency and the duty cycle from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_tpu_pwm2d_init2 118
dsrpcu_tpu_pwm2d_read 122
dsrpcu_tpu_pwm2d_req_read 125

dsrpcu_tpu_pwm2d_req_read

Syntax

```
Int16 dsrpcu_tpu_pwm2d_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   Float64* Frequency,
   Float64* Duty,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the measured duty cycle and frequency values of a specific TPU channel in request-read measurement mode.

Description

If the duty cycle is 0.0 or 1.0, the Frequency parameter is always set to zero and an error code is returned. If the measured frequency is below the lower range limit, the Frequency parameter contains a zero (for the lower range limit, refer to Range values on page 119). In this case the duty cycle is undefined because it is not clear which signal polarity the input had, and the function returns an error code. If the measured frequency is too high, the function returns unpredictable results.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_pwm2d_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the frequency and the duty cycle from the slave to the master

processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_TPU_READ_MODE_CURRENT: Read current available data
- DSRPCU_TPU_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005s).

Frequency Address where the measured frequency is stored.

Duty Pointer to an address where the measured duty cycle is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The input signal is always high or low or the frequency exceeds the range.

DSRPCU_TPU_ERR_PWM2D_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_tpu_pwm2d_init2 118
dsrpcu_tpu_pwm2d_read 122
dsrpcu_tpu_pwm2d_request 124

Pulse Width Measurement (PW2D)

Objective

To measure the pulse width of a square-wave signal.

Where to go from here

Information in this section

dsrpcu_mios_pw2d_init2

Syntax

```
void dsrpcu_mios_pw2d_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrChA,
   dsrpcu_param_t** ParamSetPtrChB,
   UInt16 ChMask,
   UInt16 ChPrescaler,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol,
   Float64 PWmax)
```

Include file

```
Dsrpcumios.h
```

Purpose

To initialize the measurement of pulse widths for a pair of MIOS channels.

Note

The functions **dsrpcu_mios_pw2d_init** and **dsrpcu_mios_f2d_init** make use of the same two channels. In consequence, either pulse-width measurement, or frequency measurement is possible for these two channels at the same time.

Description

This function initializes the measurement of pulse widths over one period.

The possible pulse width range (PW_{min} … PW_{max}) follows the following formula:

PW_{min}	2 * GP * CP / CF
PW_{max}	65534 * GP * CP / CF
GP	Global prescaler: [2 … 16]. Adjusted via dsrpcu_mios_init.
CP	Channel prescaler: [1 ... 255]. Adjusted via the ChPrescaler parameter.
CF	CPU clock frequency: 56 MHz

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrChA Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width of the first channel in the pair from the slave to the master processor. The parameter set is assigned to a specific channel and initialized when the function is being executed.

ParamSetPtrChB Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width of the second channel in the pair from the slave to the master processor. The parameter set is assigned to a specific channel and initialized when the function is being executed.

ChMask Channel pair:

- DSRPCU_MIOS_FPW2D_CH_1_2
- DSRPCU_MIOS_FPW2D_CH_3_4
- DSRPCU_MIOS_FPW2D_CH_5_6
- DSRPCU_MIOS_FPW2D_CH_7_8
- DSRPCU_MIOS_FPW2D_CH_9_10

ChPrescaler Prescaler of the MIOS channel [0 … 255].

MeasureMode Specifies the measurement mode:

- DSRPCU_MIOS_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_mios_pw2d_read.
- DSRPCU_MIOS_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. For reading the results, use dsrpcu_mios_pw2d_request and dsrpcu_mios_pw2d_req_read.

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_MIOS_INT_ENABLE
- DSRPCU_MIOS_INT_DISABLE

EdgePol Matching edge of the pulse for signal detection:

- DSRPCU_MIOS_PW2D_RISING_EDGE: The time between a rising edge and a falling edge (high time) is measured.
- DSRPCU_MIOS_PW2D_FALLING_EDGE: The time between a falling edge and a rising edge (low time) is measured.

PWmax Upper pulse width limit. PW_{max} must fit the possible period range. If the measured pulse width is greater than PW_{max} a zero is returned as the result.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7A30	Error	MIOS PW2D Init: illegal channel mask specified.	The specified value is invalid. Use one of the predefined symbols to specify the ChMask parameter.
0x7A31	Error	MIOS PW2D Init: illegal prescaler specified.	The specified value is invalid. The specified value is outside the range permitted for the ChPrescaler parameter.
0x7A32	Error	MIOS PW2D Init: illegal edge polarity parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7A33	Error	MIOS PW2D Init: illegal value for maximal pulse width specified.	The specified value is invalid. The specified value is outside the range permitted for the PWmax parameter.
0x7A34	Error	MIOS PW2D Init: channel X and Y are reserved by another function.	The specified channel pair is in use. Choose another channel pair for your function.
0x7A35	Error	MIOS PW2D Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7A36	Error	MIOS PW2D Init: illegal measurement mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x7A37	Error	MIOS PW2D Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7A0F	Error	MIOS: memory allocation error.	The memory allocation for internal data storage failed.
0x3A03	Info	MIOS PW2D Init: channel X and Y are initialized as pulse width measurement.	The MIOS channels were initialized successfully for PW2D measurement. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_mios_pw2d_read 131
dsrpcu_mios_pw2d_req_read 133
dsrpcu_mios_pw2d_request 132

dsrpcu_mios_pw2d_read

Syntax

```
Int16 dsrpcu_mios_pw2d_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* PulseWidth,
   UInt16* Status)
```

Include file

```
Dsrpcumios.h
```

Purpose

To read the most recent pulse width value and the status of the pulse width measurement in continuous measurement mode.

Description

This function reads the pulse width for one channel in the pair. If the pulse width is larger than **PWmax**, the PulseWidth parameter is set to zero.

Note

This function must be called in continuous measurement mode. Thus, it must not be called in combination with dsrpcu_mios_pw2d_request().

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the **dsrpcu_init()** function.

ParamSetPtr Start address of the allocated parameter set (**ParamSetPtrChA** or **ParamSetPtrChB**). The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (**dsrpcu_[…]_init**).

PulseWidth Address where the measured pulse width is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The input signal is always high or low or the frequency exceeds the range.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

Related topics

References

dsrpcu_mios_pw2d_init2 128
dsrpcu_mios_pw2d_req_read 133
dsrpcu_mios_pw2d_request 132

dsrpcu_mios_pw2d_request

Syntax

```
Int16 dsrpcu_mios_pw2d_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcumios.h
```

Purpose

To request the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode.

Description

This function performs a request of the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode. The requested data will be provided by the slave. The data can be read via dsrpcu_mios_pw2d_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_mios_pw2d_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_mios_pw2d_init2 128
dsrpcu_mios_pw2d_read 131
dsrpcu_mios_pw2d_req_read 133

dsrpcu_mios_pw2d_req_read

Syntax

```
Int16 dsrpcu_mios_pw2d_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   Float64* PulseWidth,
   UInt16* Status)
```

Include file

```
dsrpcumios.h
```

Purpose

To read the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode.

Description

This function reads the pulse width for one channel in the pair. If the pulse width is larger than `PWmax`, the PulseWidth parameter is set to zero.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_mios_pw2d_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_MIOS_READ_MODE_CURRENT: Read current available data
- DSRPCU_MIOS_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

PulseWidth Pointer to an address where the measured pulse width is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The pulse width parameter exceeds the maximum pulse width value.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

DSRPCU_MIOS_ERR_PW2D_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_mios_pw2d_init2 128
dsrpcu_mios_pw2d_read 131
dsrpcu_mios_pw2d_request 132

dsrpcu_tpu_pw2d_init2

Syntax

```
void dsrpcu_tpu_pw2d_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol,
   UInt16 PeriodNo,
   Float64 Tmax)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize the measurement of pulse widths on a defined TPU and defined channels.

Description

This function can measure the pulse width over several periods and calculates an average pulse width. If more than one signal is to be measured, this function must be called several times, with different channel and/or unit numbers. Each function call requires its own parameter set.

Range values

The possible period range [T_{min} … T_{max}] follows the following formula:

T_{min}	= 0x0046 * TR
T_{max}	= 0x7FFFFF * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: [2 … 448] ▪ TCR2: [8 … 120] The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period range numbers are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [s]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	2.50	0.30	3338	400	35.7
4	–	5.00	0.60	1669	200	71.4
8	8	10.0	1.20	834	100	143
14	–	17.5	2.10	477	57.1	250
–	16	20.0	2.40	417	50.0	286
–	24	30.0	3.60	278	33.3	429
28	–	35.0	4.19	238	28.6	500
–	32	40.0	4.79	209	25.0	571
42	–	52.5	6.29	159	19.0	750
56	56	70.0	8.39	119	14.3	1000
–	64	80.0	9.59	104	12.5	1140
84	–	105	12.6	79.5	9.52	1500
112	–	140	16.8	59.6	7.14	2000
–	120	150	18.0	55.6	6.67	2140
168	–	210	25.2	39.7	4.76	3000
224	–	280	33.6	29.8	3.57	4000
336	–	420	50.3	19.9	2.38	6000
448	–	560	67.1	14.9	1.79	8000

Note

The ranges are affected by the TPU workload. If the TPU executes this function on more than one channel, or executes additional functions, the maximum value may not be reached.

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is assigned to a specific unit/channel and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

MeasureMode Specifies the measurement mode:

- DSRPCU_TPU_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_tpu_pw2d_read.
- DSRPCU_TPU_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. Master polls for the requested data and reads the requested data when it is available. For reading the results, use dsrpcu_tpu_pw2d_request and dsrpcu_tpu_pw2d_req_read.

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

EdgePol Matching edge of the pulse for signal detection:

- DSRPCU_TPU_PW2D_RISING_EDGE: The time between a rising edge and a falling edge (high time) is measured.
- DSRPCU_TPU_PW2D_FALLING_EDGE: The time between a falling edge and a rising edge (low time) is measured.

PeriodNo Number of periods [1 ... 255] to be evaluated for the calculation of the average pulse width.

Tmax Upper pulse width limit. T_{max} must fit the possible period range.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F3D	Error	TPU PW2D Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7F34	Error	TPU PW2D Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F35	Error	TPU PW2D Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F36	Error	TPU PW2D Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F37	Error	TPU PW2D Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.
0x7F38	Error	TPU PW2D Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F39	Error	TPU PW2D Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7F3A	Error	TPU PW2D Init: illegal edge polarity parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7F3B	Error	TPU PW2D Init: illegal number of periods parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the PeriodNo parameter.
0x7F3C	Error	TPU PW2D Init: illegal Tmax parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the Tmax parameter.
0x7F3E	Error	TPU PW2D Init: illegal measurement mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.

ID	Type	Message[1)]	Description
0x3F05	Info	TPU PW2D Init: channel X on TPU X is initialized as pulse width measurement.	The TPU channels were initialized successfully for PW2D measurement. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_pw2d_read 139
dsrpcu_tpu_pw2d_req_read 141
dsrpcu_tpu_pw2d_request 140

dsrpcu_tpu_pw2d_read

Syntax

```
Int16 dsrpcu_tpu_pw2d_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* PulseWidth,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent pulse width value and the status of the pulse width measurement in continuous measurement mode.

Description

If the pulse width is larger than Tmax, the PulseWidth parameter is set as follows and an error code is returned:

Polarity	Input Signal Polarity	
	0	**1**
High	0	FLT_MAX
Low	FLT_MAX	0

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O

mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

PulseWidth Pointer to an address where the measured pulse width is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The pulse width parameter exceeds the maximum pulse width value.

Related topics

References

dsrpcu_tpu_pw2d_init2 135

dsrpcu_tpu_pw2d_request

Syntax

```
Int16 dsrpcu_tpu_pw2d_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To request the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode.

Description

This function performs a request of the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode. The requested data will be provided by the slave. The data can be read via dsrpcu_tpu_pw2d_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_pw2d_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_tpu_pw2d_init2 135
dsrpcu_tpu_pw2d_read 139
dsrpcu_tpu_pw2d_req_read 141

dsrpcu_tpu_pw2d_req_read

Syntax

```
Int16 dsrpcu_tpu_pw2d_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   Float64* PulseWidth,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent pulse width value and the status of the pulse width measurement in request-read measurement mode.

Description

If the pulse width is larger than `Tmax`, the PulseWidth parameter is set as follows and an error code is returned:

Polarity	Input Signal Polarity	
	0	**1**
High	0	FLT_MAX
Low	FLT_MAX	0

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_pw2d_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured pulse width from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_TPU_READ_MODE_CURRENT: Read current available data
- DSRPCU_TPU_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

PulseWidth Pointer to an address where the measured pulse width is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The pulse width parameter exceeds the maximum pulse width value.

DSRPCU_TPU_ERR_PW2D_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_tpu_pw2d_init2 135
dsrpcu_tpu_pw2d_read 139
dsrpcu_tpu_pw2d_request 140

Frequency Measurement (F2D)

Objective

To measure the frequency of a square-wave signal.

MIOS and TPU

Frequency measurement is possible via MIOS and TPU. For the characteristics of these two methods, refer to Comparing TPU and MIOS (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Where to go from here

Information in this section

dsrpcu_mios_f2d_init2

Syntax

```
void dsrpcu_mios_f2d_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrChA,
   dsrpcu_param_t** ParamSetPtrChB,
   UInt16 ChMask,
   UInt16 ChPrescaler,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol,
   Float64 Fmin)
```

Include file

```
dsrpcumios.h
```

Purpose

To initialize the frequency measurement for a pair of MIOS channels.

Note

The functions **dsrpcu_mios_pw2d_init2** and **dsrpcu_mios_f2d_init2** make use of the same two channels. In consequence, either pulse-width measurement, or frequency measurement is possible for these two channels at the same time.

Description

This function initializes the frequency measurement over one period.

The possible frequency range (f_{min} … f_{max}) follows the following formula:

f_{min}	CF / (65535 * GP * CP)
f_{max}	CF / (1 * GP * CP)
CF	CPU clock frequency: 56 MHz
GP	Global prescaler: [2 … 16]. Adjusted via dsrpcu_mios_init.
CP	Channel prescaler: [1 ... 255]. Adjusted via the ChPrescaler parameter.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrChA Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured frequency of the first channel in the pair from the slave to the master processor. The parameter set is assigned to a specific channel and initialized when the function is being executed.

ParamSetPtrChB Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured frequency of the second channel in the pair from the slave to the master processor. The parameter set is assigned to a specific channel and initialized when the function is being executed.

ChMask Channel pair:

- DSRPCU_MIOS_FPW2D_CH_1_2
- DSRPCU_MIOS_FPW2D_CH_3_4
- DSRPCU_MIOS_FPW2D_CH_5_6
- DSRPCU_MIOS_FPW2D_CH_7_8
- DSRPCU_MIOS_FPW2D_CH_9_10

ChPrescaler Prescaler of the MIOS channel [1 ... 255].

MeasureMode Specifies the measurement mode:

- DSRPCU_MIOS_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_mios_f2d_read.
- DSRPCU_MIOS_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. For reading the results, use dsrpcu_mios_f2d_request and dsrpcu_mios_f2d_req_read.

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_MIOS_INT_ENABLE
- DSRPCU_MIOS_INT_DISABLE

EdgePol Matching edge of the pulse for signal detection:

- DSRPCU_MIOS_F2D_RISING_EDGE: The time between two consecutive rising edges (high time) is measured.
- DSRPCU_MIOS_F2D_FALLING_EDGE: The time between two consecutive falling edges (low time) is measured.

Fmin Lower frequency limit. For frequencies lower than this limit, the function returns zero. Range of F_{min} depends on the prescaler settings.

Return value

Void

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7A20	Error	MIOS F2D Init: illegal channel mask specified"	The specified value is invalid. Use one of the predefined symbols to specify the ChMask parameter.
0x7A21	Error	MIOS F2D Init: illegal prescaler specified	The specified value is invalid. The specified value is outside the range permitted for the ChPrescaler parameter.
0x7A22	Error	MIOS F2D Init: illegal edge polarity parameter specified	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7A23	Error	MIOS F2D Init: illegal value for minimal frequency specified	The specified value is invalid. The specified value is outside the range permitted for the Fmin parameter.
0x7A24	Error	MIOS F2D Init: channel X and Y are reserved by another function	The specified channel pair is in use. Choose another channel pair for your function.
0x7A25	Error	MIOS F2D Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7A26	Error	MIOS F2D Init: illegal measurement mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x7A27	Error	MIOS F2D Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7F0F	Error	MIOS: memory allocation error	The memory allocation for internal data storage failed.
0x3A02	Info	MIOS F2D Init: channel X and Y are initialized as frequency measurement	The MIOS channels were initialized successfully for F2D measurement. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_mios_f2d_read 148
dsrpcu_mios_f2d_req_read 150
dsrpcu_mios_f2d_request 149

dsrpcu_mios_f2d_read

Syntax

```
Int16 dsrpcu_mios_f2d_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* Frequency,
   UInt16* Status)
```

Include file

```
Dsrpcumios.h
```

Purpose

To read the most recent frequency value and the status in continuous measurement mode.

Description

This function reads the measured frequency for one channel in the pair. Zero and an error code are returned if the frequency measured is less than a limit you specify, refer to **Fmin** on page 146.

Note

This function must be called in continuous measurement mode. Thus, it must not be called in combination with dsrpcu_mios_f2d_request().

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set (`ParamSetPtrChA` or `ParamSetPtrChB`). The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Frequency Address where the measured frequency is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The frequency parameter is below the fmin value.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

Related topics

References

dsrpcu_mios_f2d_init2 145
dsrpcu_mios_f2d_req_read 150
dsrpcu_mios_f2d_request 149

dsrpcu_mios_f2d_request

Syntax

```
Int16 dsrpcu_mios_f2d_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcumios.h
```

Purpose

To request the most recent frequency value and the status in request-read measurement mode.

Description

This function performs a request of the most recent frequency value and the status in request-read measurement mode. The requested data will be provided by the slave. The data can be read via dsrpcu_mios_f2d_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_mios_f2d_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_mios_f2d_init2 145
dsrpcu_mios_f2d_read 148
dsrpcu_mios_f2d_req_read 150

dsrpcu_mios_f2d_req_read

Syntax

```
Int16 dsrpcu_mios_f2d_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   Float64* Frequency,
   UInt16* Status)
```

Include file

```
dsrpcumios.h
```

Purpose

To read the most recent frequency value and the status in request-read measurement mode.

Description

Zero and an error code are returned if the frequency measured is less than a limit you specify, refer to **`Fmin`** on page 146.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_mios_f2d_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the **`dsrpcu_init()`** function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (**`dsrpcu_[…]_init`**).

ReadMode Specifies the read mode:

- DSRPCU_MIOS_READ_MODE_CURRENT: Read current available data
- DSRPCU_MIOS_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable **`DsRPCURequestReadTimeout`** (default: 0.005 s).

Frequency Pointer to an address where the measured frequency is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The frequency parameter is below the **Fmin** value.

DSMSC_SEMA_ACCESS_FAILED Master-slave communication error. The semaphore access for a parameter set failed.

DSMSC_SEMA_ACCESS_ERROR Master-slave communication error. There was an attempt to clear a semaphore which is not set by the master.

DSRPCU_MIOS_ERR_F2D_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_mios_f2d_init2 145
dsrpcu_mios_f2d_read 148
dsrpcu_mios_f2d_request 149

dsrpcu_tpu_f2d_init2

Syntax

```
void dsrpcu_tpu_f2d_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 MeasureMode,
   UInt16 IntEnable,
   UInt16 EdgePol,
   UInt16 PeriodNo,
   Float64 Fmin)
```

Include file

```
Dsrpcutpu.h
```

Purpose

To initialize the frequency measurement of signals on a defined TPU and defined channels.

Description

Initialization of the frequency measurement. The function can measure the frequency over several periods and calculate the average value. If more than one signal is to be measured, this function must be called several times, with

different channel and/or unit numbers. Each function call requires its own parameter set.

The possible frequency range [f_{min} … f_{max}] follows the following formula:

f_{min}	= 1 / (0x7FFFFF * TR)
f_{max}	= 1 / (0x0046 * TR)
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following frequency ranges are possible, as a function of TCR1 and TCR2:

TCR1	**TCR2**	**Tmin [µs]**	**Tmax [s]**	**1/Tmax [mHz]**	**1/Tmin [kHz]**	**Resolution [ns]**
2	–	2.50	0.30	3338	400	35.7
4	–	5.00	0.60	1669	200	71.4
8	8	10.0	1.20	834	100	143
14	–	17.5	2.10	477	57.1	250
–	16	20.0	2.40	417	50.0	286
–	24	30.0	3.60	278	33.3	429
28	–	35.0	4.19	238	28.6	500
–	32	40.0	4.79	209	25.0	571
42	–	52.5	6.29	159	19.0	750
56	56	70.0	8.39	119	14.3	1000
–	64	80.0	9.59	104	12.5	1140
84	–	105	12.6	79.5	9.52	1500
112	–	140	16.8	59.6	7.14	2000
–	120	150	18.0	55.6	6.67	2140
168	–	210	25.2	39.7	4.76	3000
224	–	280	33.6	29.8	3.57	4000
336	–	420	50.3	19.9	2.38	6000
448	–	560	67.1	14.9	1.79	8000

Note

The ranges are affected by the TPU workload. If the TPU executes this function on more than one channel, or executes additional functions, the maximum value may not be reached.
The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

MeasureMode Specifies the measurement mode:

- DSRPCU_TPU_CONT_MEAS_MODE: Continuous measurement mode. Slave provides data continuously. Master reads currently available data. For reading the results, use dsrpcu_tpu_f2d_read.

- DSRPCU_TPU_REQUEST_READ_MODE: Request-read measurement mode. Master requests data from slave. Slave provides the requested data. Master polls for the requested data and reads the requested data when it is available. For reading the results, use dsrpcu_tpu_f2d_request and dsrpcu_tpu_f2d_req_read.

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

EdgePol Matching edge of the pulse for signal detection:

- DSRPCU_TPU_F2D_RISING_EDGE: The time between two consecutive rising edges (high time) is measured.
- DSRPCU_TPU_F2D_FALLING_EDGE: The time between two consecutive falling edges (low time) is measured.

PeriodNo Number of periods [1 ... 255] to be evaluated to calculate of the average frequency.

Fmin Lower frequency limit. For frequencies lower than this limit, the function returns zero. Range of F_{min} depends on the prescaler settings.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F28	Error	TPU F2D Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F29	Error	TPU F2D Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F30	Error	TPU F2D Init: illegal fmin parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the Fmin parameter.
0x7F32	Error	TPU F2D Init: illegal measurement mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the MeasureMode parameter.
0x7F31	Error	TPU F2D Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).

ID	Type	Message[1)]	Description
0x7F2A	Error	TPU F2D Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F2B	Error	TPU F2D Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.
0x7F2C	Error	TPU F2D Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F2E	Error	TPU F2D Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7F2E	Error	TPU F2D Init: illegal edge polarity parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the EdgePol parameter.
0x7F2F	Error	TPU F2D Init: illegal number of periods parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the PeriodNo parameter.
0x3F04	Info	TPU F2D Init: channel X on TPU X is initialized as frequency measurement.	The TPU channels were initialized successfully for F2D measurement. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_f2d_read 156
dsrpcu_tpu_f2d_req_read 158
dsrpcu_tpu_f2d_request 157

dsrpcu_tpu_f2d_read

Syntax

```
Int16 dsrpcu_tpu_f2d_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* Frequency,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent frequency value and the status in continuous measurement mode.

Description

Zero and an error code are returned if the frequency measured is less than a limit you specify, refer to **Fmin** on page 155.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Frequency Pointer to an address where the measured frequency is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The frequency parameter is below the Fmin value.

Related topics

References

dsrpcu_tpu_f2d_init2 152

dsrpcu_tpu_f2d_request

Syntax

```
Int16 dsrpcu_tpu_f2d_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To request the most recent frequency value and the status in request-read measurement mode.

Description

This function performs a request of the most recent frequency value and the status in request-read measurement mode. The requested data will be provided by the slave. The data can be read via dsrpcu_tpu_digin_req_read() afterwards.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_f2d_req_read(). It must not be called in continuous measurement mode.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command.

Related topics

References

dsrpcu_tpu_f2d_init2 152
dsrpcu_tpu_f2d_read 156
dsrpcu_tpu_f2d_req_read 158

dsrpcu_tpu_f2d_req_read

Syntax

```
Int16 dsrpcu_tpu_f2d_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt8 ReadMode,
   Float64* Frequency,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the most recent frequency value and the status in request-read measurement mode.

Description

Zero and an error code are returned if the frequency measured is less than a limit you specify, refer to **`Fmin`** on page 155.

Note

This function must be called in request-read measurement mode, in combination with dsrpcu_tpu_digin_request(). It must not be called in continuous measurement mode.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the measured frequency from the slave to the master processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_TPU_READ_MODE_CURRENT: Read current available data
- DSRPCU_TPU_READ_MODE_NEW: Poll for new data from the slave as requested before

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

Frequency Pointer to an address where the measured frequency is stored.

Status Address where the current status of the PWM measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_OUT_OF_RANGE The frequency parameter is below the **`Fmin`** value.

DSRPCU_TPU_ERR_F2D_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_tpu_f2d_init2 152
dsrpcu_tpu_f2d_read 156
dsrpcu_tpu_f2d_request 157

Incremental Encoder Interface

Objective

To measure the position and the rotation speed of an incremental encoder.

Functionality

The incremental encoder interface provides functions to access incremental encoders which are connected to the controller model. You can read incremental encoder positions, detect the moving direction, and calculate the speed. The interface also provides outputs for setting new values for the encoder position.

Where to go from here

Information in this section

dsrpcu_tpu_enc_init 162
To initialize an incremental encoder on a defined TPU and defined channels.

dsrpcu_tpu_enc_write 166
To write a value to the encoder position register.

dsrpcu_tpu_enc_request 167
To request the incremental encoder position, speed and direction.

dsrpcu_tpu_enc_req_read 168
To read the incremental encoder position, the calculated speed, and the direction.

Information in other sections

Basics on Crankshaft Sensor Signals (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)
Gives information on how the motion of a crankshaft is typically measured.

dsrpcu_tpu_enc_init

Syntax

```
void dsrpcu_tpu_enc_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrChR,
   dsrpcu_param_t** ParamSetPtrChW,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 IntEnable,
   Float64 TimeOut,
   UInt16 IndexMode,
   UInt16 IntOnIndex,
   Float64 IndexPosition,
   Float64 StartPosition)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize an incremental encoder on a defined TPU and defined channels.

Description

This function uses two or three TPU channels, depending on whether an index input is used. Refer to IndexMode on page 164. The channel usage is as follows:

Channel	With Index Input	Without Index Input
Channel x	Index signal	0°-Phase signal
Channel x+1	0°-Phase signal	90°-Phase signal
Channel x+2	90°-Phase signal	Not used (free)

If more than one incremental encoder is to be controlled, this function must be called several times, with different channel and/or unit numbers. Each function call requires its own parameter set.

Range values

The possible edge distance range (T_{min} … T_{max}) follows the following formula:

T_{min}	= 0x0038 * TR
T_{max}	= 0x7FFF * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following edge distance numbers are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	2.00	1.17	855	500	35.7
4	–	4.00	2.34	427	250	71.4
8	8	8.00	4.68	214	125	143
14	–	14.0	8.19	122	71.4	250
–	16	16.0	9.36	107	62.5	286
–	24	24.0	14.0	71	41.7	429
28	–	28.0	16.4	61	35.7	500
–	32	32.0	18.7	53	31.3	571
42	–	42.0	24.6	41	23.8	750
56	56	56.0	32.8	31	17.9	1000
–	64	64.0	37.4	27	15.6	1140
84	–	84.0	49.2	20.3	11.90	1500
112	–	112	65.5	15.3	8.93	2000
–	120	120	70.2	14.2	8.33	2140
168	–	168	98.3	10.2	5.95	3000
224	–	224	131	7.6	4.46	4000
336	–	336	197	5.1	2.98	6000
448	–	448	262	3.8	2.23	8000

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrChR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current encoder position from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

ParamSetPtrChW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer a new encoder position

from the master to the slave processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 14] or [1 … 15]. The function uses two or three channels, depending on whether an index channel is provided (refer to IndexMode on page 164).

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

TimeOut If no transition occurs during the time specified by TimeOut, the speed is set to zero. The ranges of the maximum and minimum TimeOut values depend on the prescaler settings.

IndexMode Specifies the index mode of the encoder channels:

- DSRPCU_TPU_ENC_NO_INDEX: No index signal pulse is monitored. The encoder position counter starts counting at **`StartPosition`**. Two TPU channels reserved.
- DSRPCU_TPU_ENC_SET_ONCE: Only the first index signal pulse is evaluated and used to specify the IndexPosition parameter. Three TPU channels are reserved.
- DSRPCU_TPU_ENC_SET_CONT: Each index signal pulse is evaluated. The IndexPosition parameter is consecutively updated. Three TPU channels reserved.

IntOnIndex Specifies whether an interrupt is triggered by the slave when an index signal pulse occurs and the IndexPosition parameter is updated:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

IndexPosition Position that is written to the position register if an index signal pulse occurs. Range is [-8192 … 8191.75]. An index signal represents a specific encoder position.

StartPosition Initial start position of the encoder. Range is [-8192 … 8191.75].

Return value None

Messages The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F50	Error	TPU Enc Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`Unit`** parameter.
0x7F51	Error	TPU Enc Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **`Channel`** parameter.
0x7F52	Error	TPU Enc Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F53	Error	TPU Enc Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`PrescalerSel`** parameter.
0x7F54	Error	TPU Enc Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`Priority`** parameter.
0x7F55	Error	TPU Enc Init: illegal int enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`IntEnable`** parameter.
0x7F56	Error	TPU Enc Init: illegal time out parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the **`TimeOut`** parameter. The possible range depends on the prescaler settings.
0x7F57	Error	TPU Enc Init: illegal index mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`IndexMode`** parameter.
0x7F58	Error	TPU Enc Init: illegal interrupt on index parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the **`IntOnIndex`** parameter.
0x7F59	Error	TPU Enc Init: illegal index position parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the **`IndexPosition`** parameter.
0x7F5A	Error	TPU Enc Init: illegal start position parameter specified.	The specified value is invalid. The specified value is outside the range permitted for the **`StartPosition`** parameter.
0x3F08	Info	TPU Enc Init: channel %d and %d on TPU %d initialized as incr. encoder.	This message is generated if the initialization was successful, no index is created and the application was compiled using the -DDEBUG_INIT option.
0x3F09	Info	TPU Enc Init: channel %d, %d and %d on TPU %d initialized as incr. encoder.	This message is generated if the initialization was successful, an index is created and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_enc_req_read 168
dsrpcu_tpu_enc_request 167
dsrpcu_tpu_enc_write 166

dsrpcu_tpu_enc_write

Syntax

```
void dsrpcu_tpu_enc_write(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrW,
   Float64 Position)
```

Include file

```
dsrpcutpu.h
```

Purpose

To write a value to the encoder position register.

Description

This function updates the encoder position. Each encoder requires an own parameter set.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrW Start address of the allocated parameter set. The parameter set is used to transfer a new encoder position from the master processor to the slave. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Position Encoder position. The range is [-8192 ... 8191.75].

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The next free parameter set is blocked (only in FIFO mode).

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_enc_init 162
dsrpcu_tpu_enc_req_read 168
dsrpcu_tpu_enc_request 167

dsrpcu_tpu_enc_request

Syntax

```
Int16 dsrpcu_tpu_enc_request(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrR)
```

Include file

```
dsrpcutpu.h
```

Purpose

To request the incremental encoder position, speed and direction.

Description

The function requests the incremental encoder position, speed and direction. After new data is available, the slave triggers an interrupt, on condition that interrupts are enabled – see **`IntEnable`** parameter of dsrpcu_tpu_enc_init. The results can be read with dsrpcu_tpu_enc_req_read on page 168.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Address of the allocated parameter set. The parameter set is used to transfer and request the current encoder position from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_enc_init 162
dsrpcu_tpu_enc_req_read 168
dsrpcu_tpu_enc_write 166

dsrpcu_tpu_enc_req_read

Syntax

```
Int16 dsrpcu_tpu_enc_req_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrR,
   UInt16 ReadMode,
   Float64* Position,
   Float64* Speed,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the incremental encoder position, the calculated speed, and the direction.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Address of the allocated parameter set. The parameter set is used to transfer the current encoder position from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`).

ReadMode Specifies the read mode:

- DSRPCU_TPU_ENC_READ_MODE_CURRENT: dsrpcu_tpu_enc_read() returns the current available data.
- DSRPCU_TPU_ENC_READ_MODE_NEW: dsrpcu_tpu_enc_read() polls for new data from the slave. The data that was requested by dsrpcu_tpu_enc_read_request() beforehand is returned.

Note

Polling must not last longer than the timeout limit, otherwise an error occurs. The timeout limit is represented by the global variable `DsRPCURequestReadTimeout` (default: 0.005 s).

Position Address where the current encoder position is stored. The position range is [-8192 ... 8191.75].

Speed Address where the current speed (transitions per seconds) and direction are stored.

Note

Speed is set to zero, if one of the following conditions apply:

- The measured pulse width is larger than the `TimeOut`.
- The time between two consecutive measurements is larger than the upper limit of the available pulse width range, refer to dsrpcu_tpu_prescaler_set on page 42 for details on the possible pulse width ranges.
- Two consecutive measurements return the same encoder position.

For details, refer to Conditions forcing encoder speed to zero (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Status Address where the current status of the encoder position is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

DSRPCU_TPU_ENC_ERR_TIMEOUT It is not possible to calculate a valid encoder speed because of timeout.

DSRPCU_TPU_ERR_ENC_REQ_READ_TIMEOUT Polling for new data from slave lasted longer than the timeout limit.

Related topics

References

dsrpcu_tpu_enc_init..... 162
dsrpcu_tpu_enc_request..... 167
dsrpcu_tpu_enc_write..... 166

Stepper Motor

Objective

To control the actuating signal of a stepper motor.

Where to go from here

Information in this section

dsrpcu_tpu_sm_init

Syntax

```
void dsrpcu_tpu_sm_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 PrescalerSel,
   UInt16 Priority,
   UInt16 AccelSteps,
   Float64 MaxStepPeriod,
   Float64 PeriodDiff,
   Int16 Position)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize the stepper motor control on a defined TPU and defined channels.

Description

The stepper motor control always uses two channels. If more than one stepper motor is to be controlled, this function must be called several times, with different channel and unit numbers. Each function call requires its own parameter set.

Range values

The possible period range (T_{min} … T_{max}) follows the following formula:

T_{min}	= 0x0032 * TR
T_{max}	= 0x7FFF * TR
TR	TPU timer resolution: TR = TP / CF
TP	Prescaler values of the TPU time counter registers: ▪ TCR1: (2 … 448) ▪ TCR2: (8 … 120) The prescaler values can be adjusted via dsrpcu_tpu_prescaler_set.
CF	CPU clock frequency: 56 MHz

Thus, the following period range numbers are possible, as a function of TCR1 and TCR2:

TCR1	TCR2	Tmin [µs]	Tmax [ms]	1/Tmax [Hz]	1/Tmin [kHz]	Resolution [ns]
2	–	1.79	1.17	855	560	35.7
4	–	3.57	2.34	427	280	71.4
8	8	7.1	4.68	214	140	143
14	–	12.5	8.19	122	80.0	250
–	16	14.3	9.36	107	70.0	286
–	24	21.4	14.0	71	46.7	429
28	–	25.0	16.4	61	40.0	500
–	32	28.6	18.7	53	35.0	571
42	–	37.5	24.6	41	26.7	750
56	56	50.0	32.8	31	20.0	1000
–	64	57.1	37.4	27	17.5	1140
84	–	75	49.2	20.3	13.33	1500
112	–	100	65.5	15.3	10.00	2000
–	120	107	70.2	14.2	9.33	2140
168	–	150	98.3	10.2	6.67	3000
224	–	200	131	7.6	5.00	4000
336	–	300	197	5.1	3.33	6000
448	–	400	262	3.8	2.50	8000

Note

The ranges are theoretical values. In practice, the values are limited by the SC and PS modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the position update value from the master to the slave processor during run-time. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 15]. The function uses two channels. The second channel is selected automatically.

PrescalerSel Prescaler selection:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Priority of the channel. A channel of higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

AccelSteps Number of steps for accelerating and decelerating the stepper motor [0 … 13].

MaxStepPeriod Maximum period between two subsequent steps. The entered value must fit the given period range (T_{min} … T_{max}).

PeriodDiff Time difference between the acceleration and deceleration steps. The value depends on the prescaler settings. Refer to Stepper Motor Control (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Position Initial stepper motor position at the start of a simulation. Range: [-32768 … (32767-AccelSteps-1)].

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F10	Error	TPU SM Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the Unit parameter.
0x7F11	Error	TPU SM Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x7F12	Error	TPU SM Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F13	Error	TPU SM Init: illegal prescaler specified.	The specified value is invalid. Use one of the predefined symbols to specify the PrescalerSel parameter.
0x7F14	Error	TPU SM Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the Priority parameter.
0x7F15	Error	TPU SM Init: illegal number of acceleration steps.	The specified value is invalid. The specified value is outside the range permitted for the AccelSteps parameter.
0x7F16	Error	TPU SM Init: illegal stepper period specified.	The specified value is invalid. The specified value is outside the range permitted for the MaxStepPeriod parameter.
0x7F17	Error	TPU SM Init: illegal position number.	The specified value is invalid. The specified value is outside the range permitted for the Position parameter.
0x7F18	Error	TPU SM Init: illegal period difference specified.	The specified value is invalid. Use one of the predefined symbols to specify the PeriodDiff parameter.
0x3F03	Info	TPU SM Init: channel X and X on TPU X are initialized as stepper motor.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_sm_position_set..... 174

dsrpcu_tpu_sm_position_set

Syntax

```
Int16 dsrpcu_tpu_sm_position_set(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Int16 Position)
```

Include file

```
dsrpcutpu.h
```

Purpose

To set a new stepper motor position.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to transfer the position update value from the master to the slave processor during run-time. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Position Specifies the new stepper motor position. The possible range is [-32768 … (32767-AccelSteps)]. Refer to Stepper Motor Control (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_sm_init..... 171

SENT Receiver

Objective

To initialize a SENT receiver and to read messages which are transmitted via SENT protocol.

Where to go from here

Information in this section

dsrpcu_tpu_sent_rx_init2

Syntax

```
void dsrpcu_tpu_sent_rx_init2
   (dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16   Unit,
   UInt16   Channel,
   UInt16   PrescalerSel,
   UInt16   Priority,
   UInt16   NibbleCount,
   UInt16   ReceiveFIFODepth,
   Float64   TicPeriod,
   Float64   ClockDrift,
   UInt16   LowTics,
   UInt16   ZeroNibbleHighTics,
   UInt16   SyncHighTics,
   UInt16   IntMode,
   UInt16   IntNumber,
   UInt16   PauseMode,
   UInt16   FixedMsgLength)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize and configure a SENT receiver of the specified TPU channels.

Description

The function initializes and configures the SENT receiver of the specified TPU channels. Every SENT receiver requires two consecutive TPU channels.

You can configure, for example:

- The number of nibbles per SENT message, including the status nibble and the CRC nibble
- Properties of the SENT signal, like pulse durations and the clock drift

This function must be executed before the first message is read from the receiver buffer of the specified SENT receiver by calling `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`.

All SENT messages which are received before the `dsrpcu_tpu_sent_rx_init2()` function is called, are ignored.

Range values

The possible tick period range [TP_{min} … TP_{max}] and the TPU timer resolution (TR) follow these formulas:

TP_{min}	= 5 * TR
TP_{max}	= 32256 * TR / (ST * (1 + CD))
TR	TPU timer resolution (TR) = TPRS / CF
TPRS	Prescaler values of the TPU time counter registers: ▪ TCR1: 2 … 448 (13 steps) ▪ TCR2: 8 … 120 (7 steps) The prescaler values can be adjusted via `dsrpcu_tpu_prescaler_set`.
CF	CPU clock frequency: 56 MHz
ST	`LowTics` + `SyncHighTics`
CD	`ClockDrift`

If you use maximum values (`LowTics` = 15, `SynchHighTics` = 255 and `ClockDrift` = 1), the following period ranges [TP_{min} … TP_{max}] are possible:

Prescaler Values		**TP_{min} [µs]**	**TP_{max} [µs]**	**Resolution [ns]**
TCR1	**TCR2**			
2	–	0.18	2.13	35.71
4	–	0.36	4.26	71.43
8	8	0.72	8.53	142.9
14	–	1.25	14.93	250
–	16	1.43	17.06	285.7
–	24	2.15	25.60	428.6

Prescaler Values		TP_{min} [µs]	TP_{max} [µs]	Resolution [ns]
TCR1	TCR2			
28	–	2.50	29.86	500
–	32	2.86	34.13	571.4
42	–	3.75	44.80	750
56	56	5.0	59.73	1000
–	64	5.72	68.26	1143
84	–	7.5	89.60	1500
112	–	10	119.46	2000
–	120	10.72	128.00	2143
168	–	15	179.20	3000
224	–	20	238.93	4000
336	–	30	358.40	6000
448	–	40	477.86	8000

Note

- The ranges are affected by the RapidPro I/O subsystem's workload. If the I/O subsystem executes this function on more than one channel, or executes additional functions, the minimum tick period value might not be reached.
- The ranges are theoretical values. In practice, the values depend on the SC modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to read data from the SENT receiver.

Unit Specifies the TPU (time-processing unit) to be used for the SENT receiver:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel A SENT receiver allocates two consecutive channels on the same TPU. This parameter specifies the first of the channels to be used in the range 1 … 15.

PrescalerSel Specifies the prescaler of the TPU to be used as the reference:

- DSRPCU_TPU_SELECT_TCR1
- DSRPCU_TPU_SELECT_TCR2 (must not be used for engine control if used here)

Priority Specifies the priority of the specified TPU channels. A channel with higher priority is serviced more frequently:

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

NibbleCount Specifies the number of nibbles used in every SENT message including the status nibble and the CRC nibble in the range 1 … 217.

ReceiveFIFODepth Specifies the number of SENT messages which can be stored in the receive FIFO in the range 1 … 256.

TicPeriod Specifies the expected pulse length of a tick period within a SENT pulse in seconds. This is the base clock every SENT pulse is generated with.

The `TicPeriod` (TP) must fit the period range: $TP_{min} < TP < TP_{max}$.

ClockDrift Specifies the maximum allowed clock drift of the measured tick period that the SENT receiver accepts as the valid tick period. Synchronization pulses and nibble pulses are recognized as valid pulses within this range. Pulses outside this specified clock drift are recognized as invalid synchronization pulses or nibble pulses with an invalid value (< 0; > 15).

In addition the tolerance between two synchronization pulses must not exceed a factor of 1/64.

Related diagnostic information is returned during run time. If an invalid synchronization pulse is received, the current message is cut and the receiver searches for the next valid synchronization pulse.

The value is specified in the range 0 … 1, which is a percentage of the tick period, for example,

```
0:     0% clock drift
0.1:  10% clock drift
0.5:  50% clock drift
```

Note

It is not recommended to use a clock drift of 0, because:

- Every transmitter and receiver has a minimum drift, so the measured pulse length can fluctuate around a mean value.
- The maximum possible nibble pulse is `TicPeriod * (ZeroNibbleHighTics + 15 + LowTics) * (1 + ClockDrift)`. If a nibble pulse is longer than the maximum, it is recognized as a synchronization pulse. Thus, with a clock drift of 0, some fluctuating nibble pulses with a value of 15 could be interpreted as synchronization pulses.

LowTics Specifies the number of tick periods used for a SENT low pulse in the range 1 … 15.

ZeroNibbleHighTics Specifies the number of tick periods used for a SENT nibble high pulse with a value of 0 in the range 1 … 15.

SyncHighTics Specifies the number of tick periods used for a SENT synchronisation high pulse in the range 1 … 255.

IntMode Specifies interrupt generation on the master system (MicroAutoBox II/DS1007), for example, to read and transfer messages to the model triggered by an interrupt:

- DSRPCU_TPU_INT_ENABLE: An interrupt is generated on the master system after receiving a SENT message.
- DSRPCU_TPU_INT_DISABLE: No interrupt is generated on the master system after receiving a SENT message.

IntNumber Specifies the number of messages which are received and stored in the receive FIFO before an interrupt is triggered on the master system (MicroAutoBox II/DS1007) in the range 1 … 256. For example: If the number is set to 6, one interrupt is triggered on the master system after every 6th message (that is received).

This parameter takes effect only if the `IntMode` parameter is set to DSRPCU_TPU_INT_ENABLE.

PauseMode Every SENT message can contain a pause pulse transmitted at the end of the message (after the CRC nibble pulse). This optional pause pulse is generated by the transmitter, for example, to create SENT messages with a constant number of tick periods.

Specifies the pause pulse mode:

- DSRPCU_TPU_SENT_RX_PAUSE_ENABLE: Pause pulses are detected and measured. The values can be used for diagnostic purposes, for example, as follows: If you enable the `PauseMode` parameter and specify a non-zero value for the `FixedMsgLength` parameter, the receiver uses this expected message length for evaluation. If the message does not match the specified value, diagnostic information is generated.
- DSRPCU_TPU_SENT_RX_PAUSE_DISABLE: Disables the receiving of pause pulses.

The possible range of pause pulse duration [PT_{min} … PT_{max}] (in ticks) follow these formulas:

PT_{min}	= `LowTics` + `ZeroNibbleHighTics`
PT_{max}	= RoundDown (32256 * TR / (TP * (1 + CD)))
TR	TPU timer resolution (TR) = TPRS / CF
TPRS	Prescaler values of the TPU time counter registers: ▪ TCR1: 2 … 448 (13 steps) ▪ TCR2: 8 … 120 (7 steps) The prescaler values can be adjusted via `dsrpcu_tpu_prescaler_set`.
CF	CPU clock frequency: 56 MHz
TP	`TicPeriod`
CD	`ClockDrift`

FixedMsgLength Specifies the fixed message length of each message. If you specify a non-zero value for the `FixedMsgLength` parameter, the receiver uses this expected message length for evaluation and generate diagnostic information.

This parameter takes effect only if the `PauseMode` parameter is set to DSRPCU_TPU_SENT_RX_PAUSE_ENABLE.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7FE0	Error	TPU SENT Init: illegal TPU unit number specified.	The function was called with an invalid parameter. Use one of the predefined symbols to specify the `Unit` parameter.
0x7FE1	Error	TPU SENT Init: illegal channel number specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `Channel` parameter.
0x7FE2	Error	TPU SENT Init: channel X on TPU X is reserved by another function.	The specified channel is in use. Choose another channel via the `Channel for your function` parameter.
0x7FE3	Error	TPU SENT Init: illegal prescaler specified.	The function was called with an invalid parameter. Use one of the predefined symbols to specify the `PrescalerSel` parameter.
0x7FE4	Error	TPU SENT Init: illegal value for the priority specified.	The function was called with an invalid parameter. Use one of the predefined symbols to specify the `Priority` parameter.
0x7FE5	Error	TPU SENT Init: illegal value for parameter NibbleCount specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `NibbleCount` parameter.

ID	Type	Message[1)]	Description
0x7FE6	Error	TPU SENT Init: illegal value for parameter ReceiveFIFODepth specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `ReceiveFIFODepth` parameter.
0x7FE7	Error	TPU SENT Init: TicPeriod too small for selected TPU prescaler.	The function was called with a too-small value for the `TicPeriod` parameter. Specify a value which is in the range permitted for the selected TPU prescaler.
0x7FE8	Error	TPU SENT Init: illegal value for parameter clockdrift specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `ClockDrift` parameter.
0x7FE9	Error	TPU SENT Init: illegal value for parameter LowTics specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `LowTics` parameter.
0x7FEA	Error	TPU SENT Init: illegal value for parameter ZeroNibbleHighTics specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `ZeroNibbleHighTics` parameter.
0x7FEB	Error	TPU SENT Init: illegal value for parameter SyncHighTics specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `SyncHighTics` parameter.
0x7FEC	Error	TPU SENT Init: Sync pulse (LowTics + SyncHighTics) is too long.	The function was called with a too-large value for the synchronization pulse. Reduce the value for the `LowTics` parameter and/or the `SyncHighTics` parameter.
0x7FED	Error	TPU SENT Init: Zero data pulse (LowTics + ZeroNibbleHighTics) is too long.	The function was called with a too-large value for the zero data pulse. Reduce the value for the `LowTics` parameter and/or the `ZeroNibbleHighTics` parameter.
0x7FEE	Error	TPU SENT Init: illegal value for parameter IntMode specified.	The function was called with an invalid parameter. Use one of the predefined symbols to specify the `IntMode` parameter.
0x7FEF	Error	TPU SENT Init: illegal value for parameter IntNumber specified.	The function was called with an invalid parameter. Specify a value inside the range permitted for the `IntNumber` parameter.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_sent_get_rx_tic_period2 193
dsrpcu_tpu_sent_rx_receive_all2 183
dsrpcu_tpu_sent_rx_receive_most_recent2 188

dsrpcu_tpu_sent_rx_receive_all2

Syntax

```
UInt32 dsrpcu_tpu_sent_rx_receive_all2
   (dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Int8     *Data,
   UInt32    Count,
   UInt32   *Len,
   UInt32   *Diagnostic,
   UInt16   *FIFOOverflow,
   UInt16   *PausePulse)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read all new received messages and diagnostics from the receiver FIFO.

Description

The function reads all complete SENT messages received since the last read operation. The messages are saved to the `Data` buffer. The buffer must hold at least the number of expected messages defined by the `Count` parameter multiplied by the number of nibbles defined by `NibbleCount` during initialization. The number of saved messages is returned by the `Len` parameter.

You have to ensure that the data buffer specified by the `Data` parameter is large enough to hold at least the following number of Int8 values: Number of expected messages (specified by the `Count` parameter of `dsrpcu_tpu_sent_rx_receive_all()`) multiplied by the number of nibbles (specified by the `NibbleCount` parameter of `dsrpcu_tpu_sent_rx_init()`).

The `Diagnostic` parameter returns error information for every received message. You have to ensure that `Diagnostic` points to a memory area which can hold at least as many UInt32 values, as specified by the `Count` parameter.

Handling loss of data Read operations have to be executed continuously to avoid an overflow of the SENT receive FIFO. The number of messages that can be buffered between two read operations is specified by the `ReceiveFIFODepth` parameter of the `dsrpcu_tpu_sent_rx_init2` function.

The minimum necessary read frequency to avoid loss of data can be calculated from timing parameters. The minimum possible message duration multiplied by `ReceiveFIFODepth` is the maximum time between two consecutive read operations. For details, refer to Timing preconditions for receiving messages without data loss (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

If the receive FIFO is full, the oldest pulses stored in the FIFO are lost, because they are overwritten by new input data. This leads to a loss of nibbles or messages. FIFO overflow is indicated via the `FIFOOverflow` parameter during the next call of `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`.

To avoid writing more messages to the data buffer (identified by the `Data` parameter) than memory was allocated, the `Count` parameter is used. This is the maximum number of messages that are written to the data buffer. If more messages are received, the remaining messages are lost to prevent the message buffer overflowing. The return value reports the loss of data.

Handling errors If an error is detected in a SENT message (see `Diagnostic` parameter), the message is saved to the `Data` buffer with the defined number of nibbles anyway. Thus, the length of the received data array is always the length indicated by the `Len` value in messages. The error information is written to the memory area specified by the `Diagnostic` parameter. The number of written diagnostic values corresponds to the number of received `Len` messages.

- If too few nibbles are received in a message, the remaining nibbles are filled with `DSRPCU_SENT_MISSING_NIBBLE` (-128).
- If too many nibbles are received in a message, the excess nibbles are cut off, so the length of a message is always the length defined by `NibbleCount` during initialization.
- If a nibble value outside the allowed range of 0 … 15 is received, the nibble is saved to the message anyway and the diagnostic parameter reports a nibble outside the valid range.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to read data from the SENT receiver. It is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Data This buffer is filled with the received messages. The array must hold at least the number of expected messages defined by the `Count` parameter. The format of the data buffer is as follows:

Position in Data Buffer	Message / Nibble
rx_data[0]	Message 1 / Nibble 1
rx_data[1]	Message 1 / Nibble 2
…	…
rx_data[`NibbleCount` - 1] [1)]	Message 1 / Nibble `NibbleCount`
rx_data[`NibbleCount`]	Message 2 / Nibble 1
…	…

Position in Data Buffer	Message / Nibble
rx_data[2 * `NibbleCount` - 1]	Message 2 / Nibble `NibbleCount`
...	...
rx_data[(*M*-1)*`NibbleCount` + *N*-1] [2)]	Message *M* Nibble *N*

[1)] `NibbleCount` is a parameter of the `dsrpcu_tpu_sent_rx_init()` function.
[2)] This formula is valid for *M* = 1 … `Count` and *N* = 1 … `NibbleCount`. `Count` is a parameter of the `dsrpcu_tpu_sent_rx_receive_all()` function. `NibbleCount` is a parameter of the `dsrpcu_tpu_sent_rx_init` function.

If a nibble is outside its valid range, it is saved to the data buffer anyway and the error is reported by the diagnostic information. Missing nibbles are returned with a value of `DSRPCU_SENT_MISSING_NIBBLE` (-128).

Note

If you use the receive functions (`dsrpcu_tpu_sent_rx_receive_most_recent2` and `dsrpcu_tpu_sent_rx_receive_all2`) in the same real-time application for the same channel, note that the `Data` buffer is cleared during every read operation. So if you call `dsrpcu_tpu_sent_rx_receive_all2` after calling `dsrpcu_tpu_sent_rx_receive_most_recent2` ,you will only read the messages received since the execution of `dsrpcu_tpu_sent_rx_receive_most_recent2`.

Count Specifies the maximum number of messages that are written to the `Data` buffer. This value is useful to prevent writing too many messages to the allocated `Data` buffer in the real-time application.

If more messages are received than specified by the `Count` parameter, the excess messages are discarded, which is indicated by a return value of `DSRPCU_SENT_DATA_LOSS`. A return value of `DSRPCU_SENT_NO_DATA_LOSS` signals that all messages were written to the data buffer.

Len Returns the number of messages written to the `Data` buffer.

Diagnostic Returns diagnostic information for each received SENT message (UInt32 value for each received SENT message). Therefore the diagnostic vector must hold at least as many UInt32 values as specified by the `Count` parameter. The format of the diagnostic vector is as follows:

Position in Diagnostic Buffer	Meaning
diagnostic[0]	Diagnostic word (UInt32 value) of SENT message 1
diagnostic[1]	Diagnostic word (UInt32 value) of SENT message 2
...	...
diagnostic[n-1]	Diagnostic word (UInt32 value) of SENT message n

In the table above, the oldest message is SENT message 1, the newest message is SENT message n.

The SENT receiver generates a diagnostic word for every received message. The diagnostic word consists of flags indicating different message-specific status and diagnostic information. The meanings of the flags are as follows:

Bit (Flag)	Value	Information
0	1	Too many nibbles in message When too many nibbles are received in a message, the excess nibbles are ignored and the diagnostic flag reports too many nibbles. The system ensures, that every message is saved to the data buffer with the defined number of nibbles (nibble_count).
1	2	Too few nibbles in message When a message with too few nibbles is received, the missing nibbles are marked with the value of `DSRPCU_SENT_MISSING_NIBBLE` (-128) and the diagnostic flag reports missing nibbles. The system ensures, that every message is saved to the data buffer with the defined number of nibbles (nibble_count).
2	4	Nibble value is out of range 0 .. 15 When a nibble with a value <0 or >15 is received, this nibble is saved to the data buffer anyway and the diagnostic flag for a nibble out of the valid range is set.
3	8	Synchronization pulse too long (out of specified allowed clock drift). When a synchronization pulse exceeds the valid range defined by clock drift, this is reported by the diagnostic flag. The nibble values are evaluated anyway.
4	16	Synchronization pulse too short (out of specified allowed clock drift). When a synchronization pulse exceeds the valid range defined by clock drift, this is reported by the diagnostic flag. The nibble values are evaluated anyway.
5	32	Current synchronization pulse differs from the last synchronization pulse by more than a factor of 1/64. When two consecutive synchronization pulses differ by more than 1/64 of the current synchronization pulse, this is reported by the diagnostic flag.
6	64	Measured message length differs from expected (fixed) message length. If the tick periods in a received message differ from the value specified by the `FixedMsgLength` parameter, this is reported by the diagnostic flag. The message length includes the low and high ticks of all pulses of a message, including the sync pulse, the pause pulse and all nibble pulses. As preconditions for evaluation, the `PauseMode` parameter must be enabled and the value specified by `FixedMsgLength` must be set unequal to 0.
7	128	The ratio of sync pulse length to complete message length differs from the expected value. If the ratio of sync pulse length to complete message length differs by more than 1/64 (approx. 1.5 %) between the expected value (=

Bit (Flag)	Value	Information
		specified via relevant timing parameters) and the current measured value, this is reported by the diagnostic flag. This lets you observe the clock drift of a SENT transmitter while it is transmitting a SENT message. As preconditions for evaluation, the `PauseMode` parameter must be enabled and the value specified by `FixedMsgLength` must be set unequal to 0.

For example, if a message has a nibble out of range (flag 2, value 4) and the synchronization pulse is too short (flag 4, value 16), the diagnostic word value is 20. To get the information, you must evaluate the returned Diagnostic word.

PausePulses This buffer is filled with the received pause pulse values. The returned values correspond to nibble values. Example: A complete pulse duration of `LowTics` + `ZeroNibbleHighTics` results in a pause value of 0.

The array must hold at least the number of expected messages defined by the `Count` parameter. The format of the data buffer is as follows:

Position in Data Buffer	Meaning
PausePulse [0]	Pause pulse value of SENT message 1
PausePulse [1]	Pause pulse value of SENT message 2
...	...
PausePulse [n - 1]	Pause pulse value of SENT message n

In the table above, the oldest message is SENT message 1, the newest message is SENT message n.

Missing pause pulses are returned with a value of DSRPCU_SENT_MISSING_PAUSE (-32768).

FIFOOverflow Parameter that holds the address that the FIFO overflow flag will be written to. A flag value unequal 0 indicates that a FIFO overflow has occurred since the last call of the function `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`:

- DSRPCU_SENT_MASTER_BUFFER_OVERFLOW

  An overflow of the SENT receive FIFO of the master system (MicroAutoBox II/DS1007) occurred. Remedy:

  - Decrease time span between two calls of the receive functions (`dsrpcu_tpu_sent_rx_receive_all()` or `dsrpcu_tpu_sent_rx_receive_most_recent()`).
  - Increase FIFO depth via the `ReceiveFIFODepth` parameter of the `dsrpcu_tpu_sent_rx_init()` function.
- DSRPCU_SENT_SLAVE_BUFFER_OVERFLOW

  An overflow of the TPU SENT receive FIFO on the I/O subsystem (RapidPro system) occurred. Reason: The SENT Interrupt Service Routine (ISR) of the

affected SENT receiver has been interrupted or delayed by other ISRs of the RapidPro system too frequently.

- DSRPCU_SENT_SLAVE_TIMEOUT

  The length of a SENT pulse is longer than the maximum pulse length which can be measured by the SENT receiver.

Return value

DSRPCU_SENT_NO_DATA_LOSS The number of received messages did not exceed the expected number of received messages set by the `Count` parameter.

DSRPCU_SENT_DATA_LOSS The number of received messages exceeded the expected number of messages set by the `Count` parameter. At least one message was ignored to prevent the allocated message buffer overflowing. The newest messages were discarded.

Related topics

References

dsrpcu_tpu_sent_get_rx_tic_period2 193
dsrpcu_tpu_sent_rx_init2 176
dsrpcu_tpu_sent_rx_receive_most_recent2 188

dsrpcu_tpu_sent_rx_receive_most_recent2

Syntax

```
void dsrpcu_tpu_sent_rx_receive_most_recent2
   (dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Int8     *Data,
   UInt32   *Msg_count,
   UInt32   *Diagnostic,
   UInt16   *FIFOOverflow
   UInt16   *PausePulse)
```

Include file

```
dsrpcusent.h
```

Purpose

To read the most recent messages and diagnostic information from the receive FIFO.

Description

The function reads the newest complete received message from the receive FIFO. The message is saved to the `Data` buffer. The buffer must hold at least the number of nibbles specified by `NibbleCount` during initialization. The number of messages received since the last execution of a receive function is returned by the `Msg_count` parameter. If no complete message was received at all, a

message with all nibbles marked as missing nibbles `DSRPCU_SENT_MISSING_NIBBLE` (-128) is returned and the `Msg_count` parameter is set to 0.

You have to ensure that the data buffer specified by the `Data` parameter is large enough to hold at least as many Int8 values as specified by the `NibbleCount` parameter of `dsrpcu_tpu_sent_rx_init()`.

The `Diagnostic` parameter returns error information for the read message.

Handling loss of data Read operations have to be executed continuously to avoid an overflow of the SENT receive FIFO. The number of messages that can be buffered between two read operations is specified by the `ReceiveFIFODepth` parameter of the `dsrpcu_tpu_sent_rx_init2` function.

The minimum necessary read frequency to avoid loss of data can be calculated from timing parameters. The minimum possible message duration multiplied by `ReceiveFIFODepth` is the maximum time between two consecutive read operations. For details, refer to Timing preconditions for receiving messages without data loss (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

If the receive FIFO is full, the oldest pulses stored in the FIFO are lost, because they are overwritten by new input data. This leads to a loss of nibbles or messages. FIFO overflow is indicated via the `FIFOOverflow` parameter during the next call of `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`.

To avoid writing more messages to the data buffer than memory was allocated, the `Count` parameter is used. This is the maximum number of messages that are written to the data buffer. If more messages are received, the remaining messages are lost to prevent the message buffer overflowing. The return value reports the loss of data.

Handling errors If an error is detected in a SENT message (see `Diagnostic` parameter), the message is saved to the `Data` buffer with the defined number of nibbles anyway. Thus the length of the received data array is always the length specified by `NibbleCount` during initialization.

- If too few nibbles are received in a message, the remaining nibbles are filled with `DSRPCU_SENT_MISSING_NIBBLE` (-128).
- If too many nibbles are received in a message, the excess nibbles are cut off, so the length of a message is always the length defined by `NibbleCount` during initialization.
- If a nibble value outside the allowed range of 0 … 15 is received, the nibble is saved to the message anyway and the diagnostic parameter reports a nibble outside the valid range.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to read data from the SENT receiver. It is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Data This buffer is filled with the received messages. The array must hold at least the number of nibbles per message (specified with the `NibbleCount` parameter).

Position in Data Buffer	Nibble
rx_data[0]	Nibble 1
rx_data[1]	Nibble 2
…	…
rx_data[*N* - 1] [1)]	Nibble *N*

[1)] Valid if *N* = 1 … `NibbleCount`.

If a nibble is outside its valid range, it is saved to the data buffer anyway and the error is reported by the diagnostic information.

Missing nibbles are returned with a value of `DSRPCU_SENT_MISSING_NIBBLE` (-128).

Note

If you use the receive functions (`dsrpcu_tpu_sent_rx_receive_most_recent2` and `dsrpcu_tpu_sent_rx_receive_all2`) in the same real-time application for the same channel, note that the data buffer is cleared during every read operation. So if you call `dsrpcu_tpu_sent_rx_receive_all2` after calling `dsrpcu_tpu_sent_rx_receive_most_recent2`, you will only read the messages received since the execution of `dsrpcu_tpu_sent_rx_receive_most_recent2`.

Msg_count Returns the number of received messages since the last read operation. This information can be used to detect a too-long model cycle time, when the number of received messages approaches the maximum FIFO message depth (specified value of the `ReceiveFIFODepth` parameter in the `dsrpcu_tpu_sent_rx_init2` function).

Diagnostic Returns diagnostic information for the received SENT message.

The SENT receiver generates a diagnostic word for the received message. The diagnostic word consists of flags indicating different message-specific status and diagnostic information. The meanings of the flags are as follows:

Bit (Flag)	Value	Information
0	1	Too many nibbles in message When too many nibbles are received in a message, the excess nibbles are ignored and the diagnostic flag reports too many nibbles. The system ensures, that every message is saved to the data buffer with the defined number of nibbles (nibble_count).
1	2	Too few nibbles in message When a message with too few nibbles is received, the missing nibbles are marked with the value of `DSRPCU_SENT_MISSING_NIBBLE` (-128) and the diagnostic flag reports missing nibbles. The system ensures, that every message is saved to the data buffer with the defined number of nibbles (nibble_count).
2	4	Nibble value is out of range 0 .. 15 When a nibble with a value <0 or >15 is received, this nibble is saved to the data buffer anyway and the diagnostic flag for a nibble out of the valid range is set.
3	8	Synchronization pulse too long (out of specified allowed clock drift). When a synchronization pulse exceeds the valid range defined by clock drift, this is reported by the diagnostic flag. The nibble values are evaluated anyway.
4	16	Synchronization pulse too short (out of specified allowed clock drift). When a synchronization pulse exceeds the valid range defined by clock drift, this is reported by the diagnostic flag. The nibble values are evaluated anyway.
5	32	Current synchronization pulse differs from the last synchronization pulse by more than a factor of 1/64. When two consecutive synchronization pulses differ by more than 1/64 of the current synchronization pulse, this is reported by the diagnostic flag.
7	64	Measured message length differs from expected (fixed) message length. If the tick periods in a received message differ from the value specified by `FixedMsgLength`, this is reported by the diagnostic flag. The message length includes the low and high ticks of all pulses of a message, including the sync pulse, the pause pulse and all nibble pulses. As preconditions for evaluation, the `PauseMode` parameter must be enabled and the value specified by `FixedMsgLength` must be set unequal to 0.
8	128	The ratio of sync pulse length to complete message length is different to the expected value. When the ratio of sync pulse length to complete message length differs by more than 1/64 (approx. 1.5 %) between the expected value (= specified via relevant timing parameters) and the current measured value, this is reported by the diagnostic flag. This lets you

Bit (Flag)	Value	Information
		observe the clock drift of a SENT transmitter while it is transmitting a SENT message. As preconditions for evaluation, the `PauseMode` parameter must be enabled and the value specified by `FixedMsgLength` must be set unequal to 0.

For example, if a message has a nibble out of range (flag 2, value 4) and the synchronization pulse is too short (flag 4, value 16), the diagnostic word value is 20. To get the information, you must evaluate the returned Diagnostic word.

PausePulse Returns the measured pause pulse value of the received SENT message.

The returned value corresponds to a nibble value. Example: A complete pulse length of `LowTics` + `ZeroNibbleHighTics` results in a pause value of 0.

A missing pause pulse is returned with a value of DSRPCU_SENT_MISSING_PAUSE (-32768).

FIFOOverflow Parameter that holds the address that the FIFO overflow flag will be written to. A flag value unequal 0 indicates that a FIFO overflow has occurred since the last call of the function `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`:

- DSRPCU_SENT_MASTER_BUFFER_OVERFLOW

  An overflow of the SENT receive FIFO of the master system (MicroAutoBox II/DS1007) occurred. Remedy:

  - Decrease time span between two calls of the receive functions (`dsrpcu_tpu_sent_rx_receive_all()` or `dsrpcu_tpu_sent_rx_receive_most_recent()`).
  - Increase FIFO depth via the `ReceiveFIFODepth` parameter of the `dsrpcu_tpu_sent_rx_init()` function.
- DSRPCU_SENT_SLAVE_BUFFER_OVERFLOW

  An overflow of the TPU SENT receive FIFO on the I/O subsystem (RapidPro system) occurred. Reason: The SENT Interrupt Service Routine (ISR) of the affected SENT receiver has been interrupted or delayed by other ISRs of the RapidPro system too frequently.
- DSRPCU_SENT_SLAVE_TIMEOUT

  The length of a SENT pulse is longer than the maximum pulse length which can be measured by the SENT receiver.

Return value None

dsrpcu_tpu_sent_get_rx_tic_period2

Syntax

```
Float64 dsrpcu_tpu_sent_get_rx_tic_period2
   (dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To return the current tick period of the specified SENT receiver.

Description

The function returns the current tick period of the specified SENT receiver in seconds. The tick period is extracted from the last received synchronization pulse when SENT messages are read with `dsrpcu_tpu_sent_rx_receive_all2` or `dsrpcu_tpu_sent_rx_receive_most_recent2`. The value is updated with every read operation so it is constant as long as none of the receive functions mentioned above is executed.

If no receive function was executed before the tick period was read, a tick period of 0 is returned.

Range values

The possible tick period range [TP_{min} … TP_{max}] and the TPU timer resolution (TR) follow these formulas:

TP_{min}	= 5 * TR
TP_{max}	= 32256 * TR / (ST * (1 + CD))
TR	TPU timer resolution (TR) = TPRS / CF
TPRS	Prescaler values of the TPU time counter registers: ▪ TCR1: 2 … 448 (13 steps) ▪ TCR2: 8 … 120 (7 steps) The prescaler values can be adjusted via `dsrpcu_tpu_prescaler_set`.
CF	CPU clock frequency: 56 MHz
ST	`LowTics` + `SyncHighTics`
CD	`ClockDrift`

If you use maximum values (`LowTics` = 15, `SynchHighTics` = 255 and `ClockDrift` = 1), the following period ranges [TP_{min} … TP_{max}] are possible:

Prescaler Values		**TP_{min} [µs]**	**TP_{max} [µs]**	**Resolution [ns]**
TCR1	**TCR2**			
2	–	0.18	2.13	35.71
4	–	0.36	4.26	71.43

Prescaler Values		TP_{min} [µs]	TP_{max} [µs]	Resolution [ns]
TCR1	TCR2			
8	8	0.72	8.53	142.9
14	–	1.25	14.93	250
–	16	1.43	17.06	285.7
–	24	2.15	25.60	428.6
28	–	2.50	29.86	500
–	32	2.86	34.13	571.4
42	–	3.75	44.80	750
56	56	5.0	59.73	1000
–	64	5.72	68.26	1143
84	–	7.5	89.60	1500
112	–	10	119.46	2000
–	120	10.72	128.00	2143
168	–	15	179.20	3000
224	–	20	238.93	4000
336	–	30	358.40	6000
448	–	40	477.86	8000

Note

- The ranges are affected by the RapidPro I/O subsystem's workload. If the I/O subsystem executes this function on more than one channel, or executes additional functions, the minimum tick period value might not be reached.
- The ranges are theoretical values. In practice, the values depend on the SC modules used. For further information, refer to the module data sheet in the RapidPro System Hardware Reference 🕮.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to read data from the SENT receiver. It is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

Float64 Tick period of last received synchronization pulse in seconds.

Related topics

References

dsrpcu_tpu_sent_rx_init2 176
dsrpcu_tpu_sent_rx_receive_all2 183
dsrpcu_tpu_sent_rx_receive_most_recent2 188

Engine Control

General notes

Note

The engine control functions should be performed on TPU channels of highest priority. Before any other RTLib function related to engine control can be used, the dsrpcu_tpu_crank_pm_init2 RTLib function must have been called before.

Where to go from here

Information in this section

Information in other sections

Engine Control (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

Crankshaft Signal Measurement

Purpose

The RapidPro system uses the following RTLib functions for setting up crankshaft signal measurement.

Note

You must not mix TPU-related functions with I/O PLD-related functions when setting up the crankshaft signal measurement.

Where to go from here

Information in this section

TPU-Related Crankshaft Signal Measurement

Where to go from here

Information in this section

dsrpcu_tpu_crank_pm_init2

Syntax

```
void dsrpcu_tpu_crank_pm_init2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrCR,
   dsrpcu_param_t** ParamSetPtrSR,
   dsrpcu_tpu_eng_t* EngineParam,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 Priority,
   UInt16 IntEnable,
   UInt32 CfgFlags,
   UInt16 SCModuleNo,
   UInt16 SCModuleCh)
```

Include file

```
dsrpcutpu.h
```

Purpose

To set up crankshaft period measurement with angle-overflow detection and initialize the angle-angle based pulse generation.

Description

This function always reserves two channels: The first channel services the angle-overflow detection, the second channel services the crankshaft period measurement.

To use engine control on a certain TPU (A, B, or C), either this function or the **dsrpcu_tpu_crank_od_init** on page 205 function must be called, both referencing a channel of the desired TPU. Additionally, the TCR2 for the assigned TPU must be adjusted to the external clock control (refer to **dsrpcu_tpu_prescaler_set** on page 42).

Note

An external inversion of the crankshaft signal, for example, by ConfigurationDesk for RapidPro, does not have an effect on the polarity of the crankshaft signal.

The possible speed range at a MPC565 frequency of 56 MHz is [40 rpm … 20000 rpm]. The possible speed range does not depend on the TPU prescaler setting (TCR1), speed resolution, however, does:

Prescaler Value	Resolution [ns]
2	35.7
4	71.4
8	143
14	250
28	500

Prescaler Value	Resolution [ns]
42	750
56	1000
84	1500
112	2000
168	3000
224	4000
336	6000
448	8000

This function initializes the angle-angle based pulse generation, refer to Angle-Angle-Based Pulse Generation on page 249.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrCR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current speed and angle values (crankshaft values) from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

ParamSetPtrSR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current status values from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

EngineParam The passed structure defines the engine parameters, refer to **`dsrpcu_tpu_eng_t`** on page 12.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 15]. The function uses two channels. The second channel is selected automatically. Refer to Description on page 199.

Priority Priority of the channel. A channel of higher priority is serviced more frequently.

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Note

All engine control functions must have the same priority level, but **`dsrpcu_tpu_crank_pm_init2`** and **`dsrpcu_tpu_crank_od_init`** must use channels with a smaller channel number.

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_TPU_INT_ENABLE
- DSRPCU_TPU_INT_DISABLE

Note

The IntEnable parameter is not supported in this version and must be set to DSRPCU_TPU_INT_DISABLE.

CfgFlags Lets you suppress a specific number of speed measurement results (refer to **`dsrpcu_tpu_crank_pm_read`**) after a crankshaft gap occurred. The last result before the currently suppressed result is returned instead.

You can suppress up to four results. Each result refers to a specific bit of a 4-bit array. If a bit is set, the corresponding result is suppressed (see table below). Suppose you want to suppress the 1st result and the 3rd result. In this case, CfgFlags must be set to 5 (= $2^0 + 2^2$). To mimic the behavior of the RapidPro Engine Control Blockset (no suppression), CfgFlags must be set to 0.

Suppression	Bit
1st result after gap	2^0
2nd result after gap	2^1
3rd result after gap	2^2
4th result after gap	2^3

Note

To suppress results, the period angle must be set to the smallest possible value (= 360° / number of teeth).

SCModuleNo Number of a signal conditioning module. Specifies the module from which the crankshaft signal must be routed to the angle computation unit (ACU). Range is [1 … 6].

Note

For routing crankshaft signals to the angle computation unit, it is not possible to rout them from an external RapidPro SC Unit (RPSCU), but only from an SC module of a RapidPro Control Unit.

SCModuleCh Defines the channel number of the specified module. Range is [1 … 8].

Return value None

Messages The following messages are defined:

ID	Type	Message[1)]	Description
0x7900	Error	MSC: memory allocation error.	The memory allocation for internal data storage failed.
0x7903	Error	MSC: dual-port memory allocation error.	The memory allocation for internal data storage failed.
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F78	Error	TPU Crank PM Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Unit** on page 200 parameter.
0x7F79	Error	TPU Crank PM Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **Channel** on page 201 parameter.
0x7F7A	Error	TPU Crank PM Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F7B	Error	TPU Crank PM Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Priority** on page 201 parameter.
0x7F7C	Error	TPU Crank PM Init: illegal interrupt enable parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the **IntEnable** on page 201 parameter.
0x7F7D	Error	TPU Crank PM Init: illegal SC module number specified.	The specified value is invalid. The specified value is outside the range permitted for the **SCModuleNo** on page 201 parameter.
0x7F7E	Error	TPU Crank PM Init: illegal SC module channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **SCModuleCh** on page 202 parameter.
0x7F7F	Error	TPU Crank PM Init: illegal value for the number of teeth specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F80	Error	TPU Crank PM Init: illegal value for the number of gaps specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F81	Error	TPU Crank PM Init: illegal value for the number of missing teeth specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F82	Error	TPU Crank PM Init: illegal value for the ratio specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F83	Error	TPU Crank PM Init: illegal value for the angle precision specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.

ID	Type	Message[1)]	Description
0x7F84	Error	TPU Crank PM Init: illegal value for the period angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F85	Error	TPU Crank PM Init: illegal value for the start angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F86	Error	TPU Crank PM Init: illegal edge polarity specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7F87	Error	TPU Crank PM Init: illegal external trigger enable parameter specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FC8	Error	TPU Crank PM Init: illegal value for the ext trig start angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FC9	Error	TPU Crank PM Init: illegal value for the ext trig period angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FCA	Error	TPU Crank PM Init: protocol version does not support ext trig generation.	The detected protocol version does not support the external trigger signal generation feature. Please update the slave firmware application. The feature is supported since protocol version 1.4.
0x7FCB	Error	TPU Crank PM Init: illegal value for the ext trig pulse duration specified.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FCC	Error	TPU Crank PM Init: start angle is larger or equal period angle.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FCD	Error	TPU Crank PM Init: 720 is not divisible by period angle without remainder.	The specified value is invalid. Refer to dsrpcu_tpu_eng_t on page 12.
0x7FCE	Error	TPU Crank PM Init: wrong DS1601 IO PLD version. Update the RPCU IO PLD.	The detected DS1601 IO PLD version does not support the external trigger signal generation feature. Please update the DS1601 IO PLD code. The feature is supported since IO PLD code version 1.3.
0x7FF8	Error	TPU Crank PM Init: parameter CfgFlags unequal 0 not supported by installed slave firmware.	The detected DS1602 slave firmware version does not support suppression of speed measurement results after a gap. Please update the DS1602 slave firmware. The feature is supported since DS1602 slave firmware version 1.10.
0x3E90	Warning	TPU Crank PM Init: Period meas. angle larger than tooth angle: Speed meas. flags set to 0.	The suppression of speed measurement results (configurable via parameter CfgFlags) can only be used when engine parameter PeriodAngle of structure EngineParam is set to the smallest possible value (the angle of a single tooth). Otherwise, parameter CfgFlags is ignored and suppression of speed measurement results is disabled completely.
0x3F0C	Info	TPU Crank PM Init: channel X and Y on TPU Z are initialized as crank pm.	The crankshaft period measurement on channel X and Y on TPU Z was initialized successfully. This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_tpu_crank_od_init 205
dsrpcu_tpu_crank_pm_read 204
dsrpcu_tpu_eng_t 12
Engine Speed Reference Data (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_tpu_crank_pm_read

Syntax

```
Int16 dsrpcu_tpu_crank_pm_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* Speed,
   Float64* Angle,
   UInt16* EngStatus,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the speed, angle, and status of the engine.

Description

This function reads the most recently measured period and angle values of the crankshaft as well as the engine's status.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the actual speed and angle values from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function

(`dsrpcu_[…]_init`). It corresponds to the ParamSetPtrCR parameter of **`dsrpcu_tpu_crank_pm_init2`** on page 199.

Speed Address where the speed is stored.

Angle Address where the angle is stored.

EngStatus Address where the engine status is stored. The following status values are possible:

- DSRPCU_TPU_CRANK_CAM_UNSYNCHRONIZED: Crankshaft angle measurement and period measurement have not been started. Crankshaft and camshaft signals are not synchronized.
- DSRPCU_TPU_CAM_EVAL_STARTED: Speed measurement and evaluation of the camshaft signal have been started.
- DSRPCU_TPU_CAM_SYNC_STARTED: Synchronization algorithm has been started. Logging of the camshaft signal is finished.
- DSRPCU_TPU_ADJUST_CRANK_ANGEL: Crankshaft angle measurement is being adjusted (update of TCR2).
- DSRPCU_TPU_START_INJ_IGN: Generation of injection and ignition pulses is being started.
- DSRPCU_TPU_CRANK_CAM_SYNCHRONIZED: Crankshaft and camshaft signals are synchronized. Angle values are being returned.
- DSRPCU_TPU_CRANK_MEAS_STARTED: (only possible if no camshaft signal exists) Crankshaft angle measurement and period measurement have been started.

Status Address where the current status of the period measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

dsrpcu_tpu_crank_od_init 205
dsrpcu_tpu_crank_pm_init2 199

dsrpcu_tpu_crank_od_init

Syntax

```
void dsrpcu_tpu_crank_od_init(
   dsrpcu_access_t* AccessPtr,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 Priority)
```

Include file	`dsrpcutpu.h`
Purpose	To initialize crankshaft angle-overflow detection.
Description	To use engine control on a certain TPU (A, B, or C), either this function or the dsrpcu_tpu_crank_pm_init2 on page 199 function must be called, both referencing a channel of the desired TPU. Additionally, the TCR2 for the assigned TPU unit must be adjusted to the external clock control (refer to dsrpcu_tpu_prescaler_set on page 42).
I/O mapping	The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **Unit** Time-processing unit: ▪ DSRPCU_TPU_A ▪ DSRPCU_TPU_B ▪ DSRPCU_TPU_C **Channel** Channel number [1 … 16]. The function uses one channel. **Priority** Priority of the channel. A channel of higher priority is serviced more frequently. ▪ DSRPCU_TPU_CH_PRIOR_LOW ▪ DSRPCU_TPU_CH_PRIOR_MID ▪ DSRPCU_TPU_CH_PRIOR_HIGH **Note** All engine control functions must have the same priority level, but **`dsrpcu_tpu_crank_pm_init2`** and **`dsrpcu_tpu_crank_od_init`** must use channels with a smaller channel number.
Return value	None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F88	Error	TPU Crank OD Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Unit** on page 206 parameter.
0x7F89	Error	TPU Crank OD Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **Channel** on page 206 parameter.
0x7F8A	Error	TPU Crank OD Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F8B	Error	TPU Crank OD Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Priority** on page 206 parameter.
0x3F0D	Info	TPU Crank OD Init: channel X on TPU X is initialized as crank od.	This message is generated the initialization was successful and if the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_tpu_crank_pm_init2 199
dsrpcu_tpu_crank_pm_read 204

I/O PLD-Related Crankshaft Signal Measurement

Where to go from here

Information in this section

dsrpcu_crank_init 208
To initialize the measurement of crankshaft signals if the crankshaft wheel is defined via wavetable.

dsrpcu_crank_read 212
To read the speed, angle, and status of the engine.

dsrpcu_crank_reset 214
To resynchronize the crankshaft and camshaft signals.

dsrpcu_crank_init

Syntax

```
void dsrpcu_crank_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrCR,
   dsrpcu_param_t** ParamSetPtrSR,
   dsrpcu_param_t** ParamSetPtrW,
   dsrpcu_eng_t* EngineParam,
   UInt16 IntEnable,
   Int16* WheelTable,
   UInt16 SCModuleNo,
   UInt16 SCModuleCh)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To initialize the measurement of crankshaft signals if the crankshaft wheel is defined via wavetable.

Description

The dsrpcu_crank_init function lets you measure speed, angle, and rotation direction of the crankshaft. This function alone reserves one channel.

Note

An external inversion of the crankshaft signal, for example, by ConfigurationDesk for RapidPro, is not allowed. Inversion is only allowed via the RapidPro Control Unit RTI Blockset.
Crankshaft wheels without a gap are not supported.

If you use TPU-related engine control signals (e.g., injection/ignition and angle-angle-based pulses), you must also call the **dsrpcu_tpu_crank_od_init** function, once for each TPU channel used:

- Channel 1 of the TPU that is routed to the SC module/channel specified by the SCModuleNo and SCModuleCh parameters of this function.
- Channel 1 of each TPU that is used for injection/ignition or angle-angle based pulse generation.

The possible speed range at a MPC565 frequency of 56MHz is [5 … 10000 rpm]. The possible speed resolution depends on the TPU prescaler settings, refer to dsrpcu_tpu_crank_od_init.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping

to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrCR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current speed, position (angle), and direction of the rotating crankshaft from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the dsrpcu_crank_init function is being executed.

ParamSetPtrSR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current status values from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the dsrpcu_crank_init function is being executed.

ParamSetPtrW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer a synchronization restart command from the master to the slave processor (refer to the `dsrpcu_crank_reset` function). The synchronization restart command triggers engine control to repeat the complete synchronization process, regardless whether synchronization is already achieved or not).

EngineParam The passed structure defines the engine parameters, refer to `dsrpcu_tpu_eng_t` on page 12.

IntEnable Defines whether an interrupt is to be triggered by the slave when a result is available:

- DSRPCU_ADDIO_INT_ENABLE
- DSRPCU_ADDIO_INT_DISABLE

Note

The IntEnable parameter is not supported in this version and must be set to DSRPCU_ADDIO_INT_DISABLE.

WheelTable Parameter that holds the start address of the allocated wave table which defines the teeth arrangement of the used crankshaft wheel. A wheel wave table consists of 450 16-bit numbers, which corresponds to 7200 single bit values. 1 means tooth, 0 means no tooth. The angular resolution of the single bit values is 0.05° (360°/7200).

Note

The first bit (MSB) of the first 16-bit number defines the 0°-position, which is independent on the setting of the `EdgePol` parameter (refer to `dsrpcu_eng_t`).

Note

The wavetable specification approach requires a RapidPro Control Unit with MPC565 not older than board revision DS1601-12.

SCModuleNo Number of a signal conditioning module. Specifies the module from which the crankshaft signal must be routed to the angle computation unit (ACU). Range is [1 … 6].

Note

For routing crankshaft signals to the angle computation unit, it is not possible to rout them from an external RapidPro SC Unit (RPSCU), but only from an SC module of a RapidPro Control Unit.

SCModuleCh Defines the channel number of the specified module. Range is [1 … 8].

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7700	Error	ADDIO: memory allocation error	The memory allocation for internal data storage failed.
0x7750	Error	ADDIO Crank Init: illegal interrupt enable parameter specified	The specified value is invalid. Use one of the predefined symbols to specify the IntEnable parameter.
0x7751	Error	ADDIO Crank Init: illegal SC module slot specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleNo parameter.
0x7752	Error	ADDIO Crank Init: illegal SC module channel number specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleCh parameter.
0x7753	Error	ADDIO Crank Init: illegal value for the number of teeth specified	The function was called with an illegal teeth number for the crankshaft wheel.
0x7754	Error	ADDIO Crank Init: illegal value for the number of gaps specified	The function was called with an illegal number of gaps for the crankshaft wheel.
0x7755	Error	ADDIO Crank Init: illegal value for the ratio parameter specified	The function was called with an illegal ratio value.
0x7756	Error	ADDIO Crank Init: illegal value for the angle resolution specified	The function was called with an illegal angle resolution value.
0x7757	Error	ADDIO Crank Init: illegal value for the speed meas. interval specified	The function was called with an illegal value for the speed meas. interval.
0x7758	Error	ADDIO Crank Init: illegal number of teeth for speed meas. result specified	The function was called with an illegal value for the number of teeth for speed meas. result.
0x7759	Error	ADDIO Crank Init: illegal value for the start angle specified	The function was called with an illegal start angle value.

ID	Type	Message[1)]	Description
0x775A	Error	ADDIO Crank Init: illegal edge polarity specified	The function was called with an illegal edge polarity value.
0x775B	Error	ADDIO Crank Init: illegal external trigger enable parameter specified	The function was called with an illegal value for the external trigger enable parameter.
0x775C	Error	ADDIO Crank Init: illegal value for the ext trig start angle specified	The function was called with an illegal value for the external trigger start angle.
0x775D	Error	ADDIO Crank Init: illegal value for the ext trig period angle specified	The function was called with an illegal value for the external trigger period angle.
0x775E	Error	ADDIO Crank Init: illegal value for the ext trig pulse duration specified	The function was called with an illegal value for the external trigger pulse duration.
0x775F	Error	ADDIO Crank Init: start angle is larger or equal period angle	The function was called with a start angle value which is larger or equal the period angle.
0x7760	Error	ADDIO Crank Init: 720 is not divisible by period angle without remainder	The function was called with a period angle value which can not divide 720.0 without remainder.
0x7761	Error	ADDIO Crank Init: wrong DS1601 IO PLD version. Update the DS1601 IO PLD	The detected DS1601 IO PLD version does not support crankshaft wavetables. Please update the DS1601 IO PLD code. The feature is supported since IO PLD code version 1.5.
0x7762	Error	ADDIO Crank Init: illegal value for the upper speed threshold specified	The function was called with an illegal value for the upper speed threshold.
0x7763	Error	ADDIO Crank Init: illegal value for the lower speed threshold specified	The function was called with an illegal value for the lower speed threshold.
0x7764	Error	ADDIO Crank Init: lower speed threshold is larger than upper threshold	The specified value for the lower speed threshold is larger than the value for the upper speed threshold.
0x7765	Error	ADDIO Crank Init: illegal value for the reverse crank enable parameter specified	The function was called with an illegal value for the reverse crank enable parameter.
0x7766	Error	ADDIO Crank Init: forward pulse duration is too large	The function was called with a too large value for the forward pulse duration parameter.
0x7767	Error	ADDIO Crank Init: reverse pulse duration is too small	The function was called with a too small value for the reverse pulse duration parameter.
0x7768	Error	ADDIO Crank Init: wrong protocol version. Please update slave firmware	The detected protocol version does not support crankshaft wavetables. Please update the slave firmware application. The feature is supported since slave application version 2.2.
0x7769	Error	ADDIO Crank Init: Crankshaft signal evaluation already initialized	You can initialize wavetable based crankshaft signal evaluation only once.
0x776A	Error	ADDIO Crank Init: number of detected crank event types is larger than 16 (ratio parameter too small or crank pattern incorrect)	More than 16 crank event types were detected during initialization of wavetable based crankshaft signal evaluation.
0x776B	Error	ADDIO Crank Init: only one crank event type detected (ratio parameter too large or crank pattern incorrect)	Only one crank event type was detected during initialization of wavetable based crankshaft signal evaluation.
0x776C	Error	ADDIO Crank Init: at least one event in wavetable is too long	An event longer than 204.7deg. was detected during initialization of wavetable based crankshaft signal evaluation.
0x776D	Error	ADDIO Crank Init: wavetable contains more than 360 crank events	Number of events in crank wavetable is larger than 360.

ID	Type	Message[1)]	Description
0x776E	Error	ADDIO Crank Init: wavetable contains less than 2 crank events	Number of events in crank wavetable is less than 2.
0x776F	Error	ADDIO Crank Init: illegal value for upper limit of crankshaft signal errors specified	The function was called with an illegal value for the upper limit of crankshaft signal errors.
0x7770	Error	ADDIO Crank Init: Generation of external trigger signal is not supported by current version of RapidPro slave firmware	Generation of external trigger signal has to be deactivated, as this feature is not supported by current version of RapidPro slave firmware.
0x3F16	Info	ADDIO Crank Init: channel %d on module slot %d initialized for crank input	The wavetable based crankshaft signal evaluation on channel X, module slot Y was initialized successfully. This message is generated only if the application was compiled using the –DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_crank_read 212
dsrpcu_crank_reset 214
dsrpcu_crank_status_read 217
dsrpcu_tpu_eng_t 12
Engine Speed Reference Data (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_crank_read

Syntax

```
Int16 dsrpcu_crank_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrCR,
   Float64* Speed,
   Float64* Angle,
   UInt16* SyncStatus,
   UInt16* Direction,
   UInt16* Status)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To read the speed, angle, and status of the engine.

Description

This function reads the most recently measured speed, the angle of the crankshaft, the direction of the crankshaft rotation, and also the crankshaft/camshaft synchronization status.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrCR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current speed, position (angle), and direction of the rotating crankshaft from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the dsrpcu_crank_init function is being executed.

Speed Address where the speed is stored. For a reverse crankshaft rotation, speed is 0.

Angle Address where the angle is stored.

SyncStatus Address where the synchronization status is stored. The following status values are possible:

- 0: DSRPCU_CRANK_UNSYNCHRONIZED

  Crankshaft angle measurement and speed measurement have not been started. Crankshaft and camshaft signals are not synchronized.
- 1: DSRPCU_CRANK_SYNC_STARTED

  Speed measurement is enabled. Evaluation of the crankshaft and camshaft signal has been started to achieve the synchronization of these signals.
- 2: DSRPCU_CRANK_ANGLE_ADJUST

  The Time Counter Register (TCR2) is synchronized with the 0.1° pulses sent by the Angular Computation Unit (ACU). For details on TCR2 and ACU, refer to Processing the Crankshaft Signal (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).
- 3: DSRPCU_CRANK_SYNCHRONIZED_NO_INJ_IGN

  Crankshaft and camshaft signals are synchronized. Angle measurement is enabled. Generation of injection and ignition signals is still disabled.
- 4: DSRPCU_CRANK_SYNCHRONIZED

  Same as status 3, but the generation of injection and ignition signals is enabled as speed has increased above **SpeedUpperThreshold**.

- 5: DSRPCU_CRANK_SPEED_MEAS_ACTIVE

  Crankshaft and camshaft signals cannot be synchronized. Speed measurement is enabled. Angle measurement is disabled.

  Note

  Status 5 only occurs, if the camshaft signal is not evaluated.

- 15: DSRPCU_CRANK_SYNC_ERROR

  A cam signal error occurred during synchronization, or speed is greater than the maximum possible speed. All the engine control functions are disabled (pulse generation, speed and angle measurement).

  Note

  You must restart the application, or call the **`dsrpcu_crank_reset`** function.

Direction Address where the direction of the crankshaft rotation is stored. The following direction values are possible:

- 0: DSRPCU_CRANK_DIRECTION_UNDEFINED
- 1: DSRPCU_CRANK_DIRECTION_FORWARD
- 2: DSRPCU_CRANK_DIRECTION_REVERSE

Status Address where the current status of the period measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

dsrpcu_crank_init 208
dsrpcu_crank_reset 214
dsrpcu_crank_status_read 217

dsrpcu_crank_reset

Syntax

```
void dsrpcu_crank_reset(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrW)
```

Include file	`DsRPCUAddIO.h`
Purpose	To resynchronize the crankshaft and camshaft signals.
Description	This function changes the synchronization status to DSRPCU_CRANK_UNSYNCHRONIZED. The crankshaft/camshaft synchronization process is started, again.
I/O mapping	The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **ParamSetPtrW** Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the reset command from the master processor to the slave.
Return value	None
Related topics	References dsrpcu_crank_init..... 208 dsrpcu_crank_read..... 212 dsrpcu_crank_status_read..... 217

Engine Status

Purpose

The RapidPro system uses the following RTLib functions for setting up engine status measurement.

Note

You must not mix TPU-related functions with I/O PLD-related functions when setting up the engine status measurement.

Where to go from here

Information in this section

TPU-Related Engine Status

dsrpcu_tpu_eng_status_read

Syntax

```
Int16 dsrpcu_tpu_eng_status_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   dsrpcu_tpu_eng_st_t* EngStatus,
   UInt16* Status)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the engine status.

Description

This function reads all available status information of the engine in run-time.

The period measurement must be initialized, beforehand. Refer to **dsrpcu_tpu_crank_pm_init2** on page 199 function.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the current status information from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`). It corresponds to the ParamSetPtrSR parameter of **`dsrpcu_tpu_crank_pm_init2`** on page 199.

EngStatus Structure that contains all the status information of the engine. Refer to **`dsrpcu_tpu_eng_st_t`** on page 14.

Status Address where the current status of the engine status measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

I/O PLD-Related Engine Status

dsrpcu_crank_status_read

Syntax

```
Int16 dsrpcu_crank_status_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrSR,
   dsrpcu_eng_st_t* EngStatus,
   UInt16* Status)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To read the engine error status.

Description

This function reads all available status information of the engine in run-time.

The crank signal measurement must be initialized, beforehand, refer to dsrpcu_crank_init on page 208.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrSR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current status values from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the dsrpcu_crank_init function is being executed.

EngStatus Structure that contains all the status information of the engine. Refer to **`dsrpcu_eng_st_t`** on page 23.

Status Address where the current status of the engine status measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

Camshaft Signal Measurement

Purpose

The RapidPro system uses the following RTLib functions for setting up camshaft signal measurement.

Note

You must not mix TPU-related functions with I/O PLD-related functions when setting up the camshaft signal measurement.

Where to go from here

Information in this section

TPU-Related Camshaft Signal Measurement

Where to go from here

Information in this section

dsrpcu_tpu_cam_init3

Syntax

```
void dsrpcu_tpu_cam_init3(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrR,
   dsrpcu_param_t** ParamSetPtrW,
   dsrpcu_tpu_cam_t* CamPattern,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 Priority,
   UInt16 SyncMode,
   UInt16 PhaseMeas,
   UInt16 PhaseMeasSyncMode,
   UInt16 PhaseStartPos,
   Float64 PhaseShiftAngleMax,
   Float64 PhaseShiftAngleMin,
   UInt16 PhaseShiftErrorMax,
   UInt16 PhaseShiftErrorReset,
   UInt16 SCModuleNo,
   UInt16 SCModuleCh)
```

Include file

```
dsrpcutpu.h
```

Purpose

To initialize camshaft evaluation.

Description

If more than one camshaft channel is desired, call this function several times, once for each of the available camshaft channels. Up to 16 camshaft channels can be initialized.

Note

If the crankshaft wheel has no gap, the following rules apply:

- Only a maximum of four camshaft wheels can be evaluated for synchronization purposes.
- All camshaft wheels used for synchronization must not have more than one cam.
- If camshaft phase measurement is enabled, in addition, the following applies:
  - DSRPCU_TPU_CAM_SYNC_ALWAYS option must not be selected as the synchronization mode.
  - All camshaft wheels must be shifted to initial position (including the ones only used for camshaft phase measurement).

Note

An external inversion of the camshaft signal, for example, by ConfigurationDesk for RapidPro, is not allowed. Inversion is only allowed via the RapidPro Control Unit RTI Blockset.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current camshaft phase values from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

ParamSetPtrW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the status of the camshaft wheel, whether it is in initial position, from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

CamPattern Defines the segmentation of the camshaft wheel, refer to **`dsrpcu_tpu_cam_t`** on page 16.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

Priority Priority of the channel. A channel of higher priority is serviced more frequently.

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID

- DSRPCU_TPU_CH_PRIOR_HIGH

Note

All engine control functions must have the same priority level, but **dsrpcu_tpu_crank_pm_init2** on page 199 and **dsrpcu_tpu_crank_od_init** on page 205 must use channels with a smaller channel number.

SyncMode Specifies the behavior of the synchronization algorithm if two or more camshaft signals are evaluated and, for a specific camshaft signal, no camshaft marker is detected within the current evaluation segment:

- DSRPCU_TPU_CAM_EDGE_MANDATORY:

  It is mandatory that a camshaft marker is detected for each camshaft signal. If, for a certain camshaft signal, no camshaft marker is detected within the current evaluation segment, the next evaluation segment of the crankshaft wheel is taken into account, and so on.
- DSRPCU_TPU_CAM_NO_EDGE_REQUIRED:

  It is not mandatory that a camshaft marker is detected. The synchronization algorithm tries to synchronize by evaluating the other camshaft signals. This may lead to a quicker synchronization.

Note

If SyncMode is set to DSRPCU_TPU_CAM_NO_EDGE_REQUIRED for one camshaft signal, this parameter must be set to DSRPCU_TPU_CAM_NO_EDGE_REQUIRED, too, in the `dsrpcu_tpu_cam_init2()` function calls of all the other camshaft signals.

NOTICE

Risk of severe engine damage.

A faulty synchronization may occur when a camshaft signal fails or is disrupted. A faulty synchronization can cause severe damage of the internal combustion engine.

- Check the connections and the operability of the camshaft sensors.

To mimic the behavior of the RapidPro Engine Control Blockset, SyncMode must be set to DSRPCU_TPU_CAM_EDGE_MANDATORY.

PhaseMeas Measurement mode of the camshaft phase shift:

- DSRPCU_TPU_CAM_PHASE_MEAS_ENABLE
- DSRPCU_TPU_CAM_PHASE_MEAS_DISABLE

PhaseMeasSyncMode Specifies the synchronization behavior with respect to a camshaft phase shift (refer to Synchronization Details on Parameter-Based Crankshaft Wheel Specifications (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)):

- DSRPCU_TPU_CAM_SYNC_LOCK_POS: Synchronization is possible only if there is no camshaft phase shift during initialization.

  The **dsrpcu_tpu_cam_phase_status_write** on page 228 function indicates whether the camshaft wheel is in initial position.
- DSRPCU_TPU_CAM_SYNC_ALWAYS: Synchronization is even possible if there is a camshaft phase shift during initialization. However, general limitations have to be considered.
- DSRPCU_TPU_CAM_SYNC_IGNORE: The camshaft wheel position is ignored in synchronization. The measurement of a camshaft phase shift is nonetheless possible.

Not evaluated if the PhaseMeas parameter is set to DSRPCU_TPU_CAM_PHASE_MEAS_DISABLE.

PhaseStartPos Specifies the camshaft phase shift status during initialization:

- DSRPCU_TPU_CAM_PHASE_POS_INIT: The camshaft phase shift is zero.
- DSRPCU_TPU_CAM_PHASE_POS_SHIFTED: There is a camshaft phase shift, which has to be offset before synchronization can take place.

Not evaluated if the SyncMode parameter is set to DSRPCU_TPU_CAM_SYNC_ALWAYS, and if the PhaseMeas parameter is set to DSRPCU_TPU_CAM_PHASE_MEAS_DISABLE.

PhaseShiftAngleMax Specifies the maximum possible camshaft phase shift. The range is [> -120° ... 120°]. The following rule applies:

Note

PhaseShiftAngleMax > PhaseShiftAngleMin

PhaseShiftAngleMin Specifies the minimum possible camshaft phase shift. The range is [-120°... < 120°].

PhaseShiftErrorMax Specifies the maximum number of consecutive invalid camshaft phase shift measurement values. If the number of accumulated errors is greater than PhaseShiftErrorMax, the measurement is stopped and resynchronized to the angle counter (TRC2). A measurement value is invalid if it exceeds the range [PhaseShiftAngleMin ... PhaseShiftAngleMax]. The range of the error counter is (1 ... 100), the default value is 3.

PhaseShiftErrorReset Specifies the number of consecutive valid camshaft phase shift measurement values that are required before the error counter is reset to 0. A measurement value is valid if it fits the range

[PhaseShiftAngleMin ... PhaseShiftAngleMax]. The range of the counter is (1 ... 100), the default value is 2.

SCModuleNo Number of a signal conditioning module. Specifies the module from which the camshaft signal must be routed to the angle computation unit (ACU). Range is [1 … 6]. SCModuleNo is evaluated only if the number of crankshaft gaps is zero.

Note

For routing camshaft signals to the angle computation unit, it is not possible to rout them from an external RapidPro SC Unit (RPSCU), but only from an SC module of a RapidPro Control Unit.

SCModuleCh Defines the channel number of the specified module. Range is [1 … 8]. SCModuleCh is evaluated only if the number of crankshaft gaps is zero.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7F90	Error	TPU Cam Init: illegal TPU unit number specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Unit** on page 221 parameter.
0x7F91	Error	TPU Cam Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **Channel** on page 221 parameter.
0x7F92	Error	TPU Cam Init: channel X on TPU X was reserved by another function.	The specified channel is in use. Choose another channel for your function.
0x7F93	Error	TPU Cam Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the **Priority** on page 221 parameter.
0x7F94	Error	TPU Cam Init: illegal phase measurement mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the **PhaseMeas** on page 222 parameter.
0x7F95	Error	TPU Cam Init: illegal phase measurement synchronization mode specified.	The specified value is invalid. Use one of the predefined symbols to specify the **PhaseMeasSyncMode** on page 223 parameter.
0x7F96	Error	TPU Cam Init: illegal value for the vector size specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F97	Error	TPU Cam Init: illegal value for the X. element of the cam vector specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F98	Error	TPU Cam Init: illegal value for the tolerance specified.	The specified tolerance value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F99	Error	TPU Cam Init: illegal edge polarity parameter specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.

ID	Type	Message[1)]	Description
0x7F9A	Error	TPU Cam Init: illegal start signal polarity parameter specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F9B	Error	TPU Cam Init: illegal value for the segment start specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F9C	Error	TPU Cam Init: illegal value for the segment end specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7F9D	Error	TPU Cam Init: illegal start position specified.	The specified value is invalid. Use one of the predefined symbols to specify the **PhaseStartPos** on page 223 parameter.
0x7F9E	Error	TPU Cam Init: illegal SC module number.	The specified value is invalid. The specified value is outside the range permitted for the **SCModuleNo** on page 224 parameter.
0x7F9F	Error	TPU Cam Init: illegal SC module channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the **SCModuleCh** on page 224 parameter.
0x7FA0	Error	TPU Cam Init: crank wheels without gap: to much cam wheels specified.	The specified value is invalid. If the crankshaft wheel has no gaps you can only specify a maximum of four camshaft wheels.
0x7FA1	Error	TPU Cam Init: illegal phase measurement and sync mode combination specified.	The specified value is invalid. If the crankshaft has no gaps and PhaseMeas is enabled, the DSRPCU_TPU_CAM_SYNC_ALWAYS option must not be selected as SyncMode.
0x7FA2	Error	TPU Cam Init: crank wheels without gap: illegal vector size specified.	The specified value is invalid. If the crankshaft has no gaps the value of the **vector size** parameter must be 1 or 2, depending on the edge polarity parameter (refer to dsrpcu_tpu_cam_t).
0x7FA3	Error	TPU Cam Init: illegal interrupt enable parameter specified.	The specified value is invalid. Refer to dsrpcu_tpu_cam_t on page 16.
0x7FBD	Error	TPU Cam Init: protocol version does not support no edge detection required.	The detected DS1602 slave firmware version does not support value DSRPCU_TPU_CAM_NO_EDGE_REQUIRED for parameter SyncMode. Please update the DS1602 slave firmware. The feature is supported since DS1602 slave firmware version 1.10.
0x7FBE	Error	TPU Cam Init: illegal synchronization mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the SyncMode parameter.
0x7FBF	Error	TPU Cam Init: parameter SyncMode has to be identical for all cam wheels.	The value assigned to parameter SyncMode must be identical for all calls of function dsrpcu_tpu_cam_init2().
0x7600	Error	TPU Cam Init: phase shift meas. min and max values are not supported by the protocol version.	Specifying a phase shift validity range is not supported by the application firmware of your RapidPro system.
0x7601	Error	TPU Cam Init: phase shift meas. error count value is not supported by the protocol version.	The error counter for the phase shift measurement is not supported by the application firmware of your RapidPro system.
0x7602	Error	TPU Cam Init: phase shift meas. reset error count value is not supported by the protocol version.	An error counter reset is not supported by the application firmware of your RapidPro system.
0x7603	Error	TPU Cam Init: illegal value for the phase shift max value specified.	The function was called with an illegal value for the PhaseShiftAngleMax parameter.

ID	Type	Message[1)]	Description
0x7604	Error	TPU Cam Init: illegal value for the phase shift min value specified.	The function was called with an illegal value for the PhaseShiftAngleMin parameter.
0x7605	Error	TPU Cam Init: phase shift min value is larger than phase shift max value.	The value of the PhaseShiftAngleMin parameter is larger than the value of the PhaseShiftAngleMax parameter.
0x7606	Error	TPU Cam Init: illegal value for the maximum error count value specified.	The function was called with an illegal value for the PhaseShiftErrorMax parameter.
0x7607	Error	TPU Cam Init: illegal value for the reset error count value specified.	The function was called with an illegal value for the PhaseShiftErrorReset parameter.
0x7608	Error	TPU Cam Init: wrong protocol version. Please update the application firmware.	The slave firmware does not support the actual camshaft functionality.
0x3F0E	Info	TPU Cam Init: channel X on TPU X is initialized as camshaft input.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

Basics

Possible Crankshaft Wheels (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_tpu_cam_phase_read2 226
dsrpcu_tpu_cam_phase_status_write 228

dsrpcu_tpu_cam_phase_read2

Syntax

```
Int16 dsrpcu_tpu_cam_phase_read2(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64* Phase,
   UInt16* Status,
   UInt32* ErrorCount)
```

Include file

```
dsrpcutpu.h
```

Purpose

To read the measured camshaft phase.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the actual status from the slave to the master processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`). It corresponds to the ParamSetPtrR parameter of **`dsrpcu_tpu_cam_init3`** on page 220.

Phase Address where the most recent result of the camshaft phase measurement is stored. Range of the camshaft phase is [-120.0° … 120.0°].

Status Address where the current status of the cam phase measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

ErrorCount Address where the current phase shift error count is stored. The value is incremented if the phase measurement detects a value that exceeds the range [PhaseShiftAngleMin ... PhaseShiftAngleMax] which is defined in **`dsrpcu_tpu_cam_init3`** on page 220. If a NULL pointer is passed to the function the value is not incremented.

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

dsrpcu_tpu_cam_phase_status_write

Syntax

```
Int16 dsrpcu_tpu_cam_phase_status_write(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt16 PhaseStatus)
```

Include file

```
dsrpcutpu.h
```

Purpose

To write the current phase status of a camshaft wheel.

Description

The phase status of a camshaft wheel can be described as follows: Camshaft wheel is initial position, or it is not.

For synchronization the software must know the camshaft start position at initialization. If there is a camshaft phase shift, it is to be set off before synchronization can take place. Refer to Synchronization With Camshaft Phase Shift (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the actual status from the master to the slave processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`). It corresponds to the ParamSetPtrW parameter of **`dsrpcu_tpu_cam_init3`** on page 220.

PhaseStatus Status of the camshaft wheel:

- DSRPCU_TPU_CAM_PHASE_POS_INIT: Initial position.
- DSRPCU_TPU_CAM_PHASE_POS_SHIFTED: Other position.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_cam_init3 220
dsrpcu_tpu_cam_phase_read2 226

I/O PLD-Related Camshaft Signal Measurement

Where to go from here

Information in this section

dsrpcu_cam_init 229
To initialize the camshaft signal measurement.

dsrpcu_cam_phase_shift_init 232
To initialize the camshaft phase shift measurement.

dsrpcu_cam_phase_shift_read 236
To read the measured camshaft phase.

dsrpcu_cam_phase_shift_status_write 237
To write the current phase status of a camshaft wheel.

dsrpcu_cam_init

Syntax

```
void dsrpcu_cam_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrW,
   dsrpcu_cam_t* CamPattern,
   UInt16 Unit,
   UInt16 Channel,
   UInt16 SyncMode,
   UInt16 SCModuleNo,
   UInt16 SCModuleCh)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To initialize the camshaft signal measurement.

Description

If more than one camshaft channel is desired, call this function several times, once for each of the available camshaft channels. Up to 4 camshaft channels can be initialized.

Note

An external inversion of the camshaft signal, for example, by ConfigurationDesk for RapidPro, is not allowed. Inversion is only allowed via the RapidPro Control Unit RTI Blockset.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the command for updating the characteristics of the camshaft wheel. The parameter set is assigned to a specific channel/unit and initialized when the dsrpcu_cam_init function is being executed.

CamPattern Defines the segmentation of the camshaft wheel, refer to **`dsrpcu_cam_t`** on page 24.

Unit Time-processing unit:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

Channel Channel number [1 … 16]. The function uses one channel.

SyncMode This parameter is reserved for future use. It must be set to 0.

SCModuleNo Number of a signal conditioning module. Specifies the module from which the camshaft signal must be routed to the angle computation unit (ACU). Range is [1 … 6]. SCModuleNo is evaluated only if the number of crankshaft gaps is zero.

Note

For routing camshaft signals to the angle computation unit, it is not possible to rout them from an external RapidPro SC Unit (RPSCU), but only from an SC module of a RapidPro Control Unit.

SCModuleCh Defines the channel number of the specified module. Range is [1 … 8]. SCModuleCh is evaluated only if the number of crankshaft gaps is zero.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7780	Error	ADDIO Cam Init: crank channel has not been initialized yet	Camshaft initialization function has been called prior to crankshaft initialization.
0x7781	Error	ADDIO Cam Init: cam channel has already been initialized	Camshaft channel has already been initialized.
0x7782	Error	ADDIO Cam Init: illegal value for the vector size specified	Illegal camshaft vector size specified.
0x7783	Error	ADDIO Cam Init: illegal value for the %d. element of cam vector specified	The specified value of the camshaft vector is invalid.
0x7784	Error	ADDIO Cam Init: illegal value for the tolerance specified	The specified value is invalid.
0x7785	Error	ADDIO Cam Init: illegal edge polarity parameter specified	The specified value is invalid.
0x7786	Error	ADDIO Cam Init: illegal start signal polarity parameter specified	The specified value is invalid.
0x7787	Error	ADDIO Cam Init: illegal SC module number specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleNo parameter.
0x7788	Error	ADDIO Cam Init: illegal SC module channel number specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleCh parameter.
0x7789	Error	ADDIO Cam Init: illegal value for parameter IntEnable specified	The specified value is invalid. The specified value is outside the range permitted for the IntEnable parameter.
0x778A	Error	ADDIO Cam Init: illegal TPU unit specified	The specified value is invalid. The specified value is outside the range permitted for the Unit parameter.
0x778B	Error	ADDIO Cam Init: illegal TPU channel specified	The specified value is invalid. The specified value is outside the range permitted for the Channel parameter.
0x778C	Error	ADDIO Cam Init: specified TPU channel already reserved by another function	The specified TPU channel has already been reserved by another function.

ID	Type	Message[1)]	Description
0x778D	Error	ADDIO Cam Init: specified cam wheel tolerance is larger than 127 cam edges	The specified value is invalid. The specified value is outside the range permitted for the Tolerance parameter.
0x3705	Info	ADDIO Cam Init: module channel %d on slot %d initialized	Camshaft signal evaluation (wavetable based engine control) on channel X, module slot Y was initialized successfully. This message is generated only if the application was compiled using the –DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_cam_phase_shift_init 232
dsrpcu_cam_phase_shift_read 236
dsrpcu_cam_phase_shift_status_write 237

dsrpcu_cam_phase_shift_init

Syntax

```
void dsrpcu_cam_phase_shift_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtrR,
   dsrpcu_param_t** ParamSetPtrCW,
   dsrpcu_param_t** ParamSetPtrSW,
   Float64 PhaseShiftAngleMax,
   Float64 PhaseShiftAngleMin,
   UInt16 PhaseShiftErrorMax,
   UInt16 PhaseShiftErrorReset,
   UInt16 PhaseMeasSyncMode,
   UInt16 PhaseStartPos,
   UInt16 SCModuleNo,
   UInt16 SCModuleCh)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To initialize the camshaft phase shift measurement.

Description

If more than one camshaft channel is desired, call this function several times, once for each of the available camshaft channels. Up to 4 camshaft channels can be initialized.

Note

An external inversion of the camshaft signal, for example, by ConfigurationDesk for RapidPro, is not allowed. Inversion is only allowed via the RapidPro Control Unit RTI Blockset.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current phase shift in ° (degree) from the slave to the master processor.

ParamSetPtrCW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the command for resetting the camshaft phase shift measurement from the master to the slave processor.

ParamSetPtrSW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current phase shift status of the camshaft wheel, whether it is in initial position or not, from the master to the slave processor.

PhaseMeasSyncMode Specifies the synchronization behavior with respect to a camshaft phase shift (refer to Synchronization Details on Parameter-Based Crankshaft Wheel Specifications (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)):

- DSRPCU_ADDIO_CAM_SYNC_LOCK_POS: Synchronization is possible only if there is no camshaft phase shift during initialization.

  The **`dsrpcu_tpu_cam_phase_status_write`** on page 228 function indicates whether the camshaft wheel is in initial position.
- DSRPCU_ADDIO_CAM_SYNC_ALWAYS: Synchronization is even possible if there is a camshaft phase shift during initialization. However, general limitations have to be considered.
- DSRPCU_ADDIO_CAM_SYNC_IGNORE: The camshaft wheel position is ignored in synchronization. The measurement of a camshaft phase shift is nonetheless possible.

PhaseStartPos Specifies the camshaft phase shift status during initialization:

- DSRPCU_ADDIO_CAM_PHASE_POS_INIT: The camshaft phase shift is zero.
- DSRPCU_ADDIO_CAM_PHASE_POS_SHIFTED: There is a camshaft phase shift, which has to be offset before synchronization can take place.

Not evaluated if the PhaseMeasSyncMode parameter is set to DSRPCU_ADDIO_CAM_SYNC_ALWAYS or DSRPCU_ADDIO_CAM_SYNC_IGNORE.

PhaseShiftAngleMax Specifies the maximum possible camshaft phase shift. The range is [> -120° ... 120°]. The following rule applies:

Note

PhaseShiftAngleMax > PhaseShiftAngleMin

PhaseShiftAngleMin Specifies the minimum possible camshaft phase shift. The range is [-120°... < 120°].

PhaseShiftErrorMax Specifies the maximum number of consecutive invalid camshaft phase shift measurement values. If the number of accumulated errors is greater than PhaseShiftErrorMax, the measurement is stopped and resynchronized to the angle counter (TRC2). A measurement value is invalid if it exceeds the range [PhaseShiftAngleMin ... PhaseShiftAngleMax]. The range of the error counter is [1 ... 100], the default value is 3.

PhaseShiftErrorReset Specifies the number of consecutive valid camshaft phase shift measurement values that are required before the error counter is reset to 0. A measurement value is valid if it fits the range [PhaseShiftAngleMin ... PhaseShiftAngleMax]. The range of the counter is [1 ... 100], the default value is 2.

SCModuleNo Number of a signal conditioning module. Specifies the module from which the camshaft signal must be routed to the angle computation unit (ACU). Range is [1 … 6]. SCModuleNo is evaluated only if the number of crankshaft gaps is zero.

Note

For routing camshaft signals to the angle computation unit, it is not possible to rout them from an external RapidPro SC Unit (RPSCU), but only from an SC module of a RapidPro Control Unit.

SCModuleCh Defines the channel number of the specified module. Range is [1 … 8]. SCModuleCh is evaluated only if the number of crankshaft gaps is zero.

Return value None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x77A0	Error	ADDIO Cam Phase Init: crank channel has not been initialized yet	Function dsrpcu_crank_init() has not been called prior call of this function.
0x77A1	Error	ADDIO Cam Phase Init: channel %d on slot %d is not initialized as camshaft input	The specified channel has not been initialized as camshaft input.
0x77A2	Error	ADDIO Cam Phase Init: channel %d on slot %d has already been reserved for cam phase meas	The specified channel has already been reserved for camshaft phase measurement.
0x77A3	Error	ADDIO Cam Phase Init: illegal value for parameter PhaseMeasAngleMax specified	The specified value is invalid. The specified value is outside the range permitted for the PhaseMeasAngleMax parameter.
0x77A4	Error	ADDIO Cam Phase Init: illegal value for parameter PhaseMeasAngleMin specified	The specified value is invalid. The specified value is outside the range permitted for the PhaseMeasAngleMin parameter.
0x77A5	Error	ADDIO Cam Phase Init: illegal phase measurement synchronization mode specified	The specified value is invalid. The specified value is outside the range permitted for the PhaseMeasSyncMode parameter.
0x77A6	Error	ADDIO Cam Phase Init: illegal start position specified	The specified value is invalid. The specified value is outside the range permitted for the PhaseStartPos parameter.
0x77A7	Error	ADDIO Cam Phase Init: illegal SC module slot specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleNo parameter.
0x77A8	Error	ADDIO Cam Phase Init: illegal SC module channel number specified	The specified value is invalid. The specified value is outside the range permitted for the SCModuleCh parameter.
0x3706	Info	ADDIO Cam Phase Init: channel number %d on slot %d initialized	Camshaft phase measurement (wavetable based engine control) on channel X, module slot Y was initialized successfully. This message is generated only if the application was compiled using the –DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_cam_init..... 229
dsrpcu_cam_phase_shift_read..... 236
dsrpcu_cam_phase_shift_status_write..... 237

dsrpcu_cam_phase_shift_read

Syntax

```
Int16 dsrpcu_cam_phase_shift_read(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrR,
   Float64* Phase,
   UInt16* Status,
   UInt16* ErrorCount)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To read the measured camshaft phase.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrR Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current phase shift in ° (degree) from the slave to the master processor.

Phase Address where the most recent result of the camshaft phase measurement is stored. Range of the camshaft phase is [-120.0° … 120.0°].

Status Address where the current status of the cam phase measurement is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

ErrorCount Address where the current phase shift error count is stored. The value is incremented if the phase measurement detects a value that exceeds the range [PhaseShiftAngleMin ... PhaseShiftAngleMax] which is defined in **`dsrpcu_cam_init`** on page 229. If a NULL pointer is passed to the function the value is not incremented.

Return value

DSRPCU_NO_ERROR No error occurred.

Related topics

References

dsrpcu_cam_phase_shift_status_write

Syntax

```
Int16 dsrpcu_cam_phase_shift_status_write(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtrSW,
   UInt16 PhaseStatus)
```

Include file

```
DsRPCUAddIO.h
```

Purpose

To write the current phase status of a camshaft wheel.

Description

The phase status of a camshaft wheel can be described as follows: Camshaft wheel is initial position, or it is not.

For synchronization the software must know the camshaft start position at initialization. If there is a camshaft phase shift, it is to be set off before synchronization can take place. Refer to Synchronization With Camshaft Phase Shift (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtrSW Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the current phase shift status of the camshaft wheel, whether it is in initial position or not, from the master to the slave processor. The parameter set is assigned to a specific

channel/unit and initialized when the dsrpcu_cam_phase_shift_init function is being executed.

PhaseStatus Status of the camshaft wheel:

- DSRPCU_ADDIO_CAM_PHASE_POS_INIT:
  Camshaft wheel is in initial position.
- DSRPCU_ADDIO_CAM_PHASE_POS_SHIFTED:
  Camshaft wheel is not in initial position.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_cam_init..... 229
dsrpcu_cam_phase_shift_init..... 232
dsrpcu_cam_phase_shift_read..... 236

Injection and Ignition Pulse Generation

Purpose

The RapidPro system uses the following RTLib functions for specifying injection and ignition pulse sequences.

Tip

For injection and ignition pulse generation, it does not matter whether the crankshaft/camshaft signal measurement is set up via TPU- or I/O PLD-related functions.

Where to go from here

Information in this section

Information in other sections

Generating Injection and Ignition Pulses (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)
To implement the generation of injection and ignition pulses.

dsrpcu_tpu_inj_ign_init

Syntax

```
void dsrpcu_tpu_inj_ign_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   dsrpcu_tpu_inj_t* InjPattern,
   dsrpcu_tpu_ign_t* IgnPattern,
   UInt16 InjUnit,
   UInt16 IgnUnit,
   UInt16 InjChannel,
   UInt16 IgnChannel,
   UInt16 InjPriority,
   UInt16 IgnPriority,
   UInt16 CylinderNo,
   Float64 TDCAngle,
   UInt16 InjOverlapMode,
   UInt16 IgnOverlapMode,
   UInt16 GateMode,
   UInt16 InjPolarity,
   UInt16 IgnPolarity)
```

Include file

```
dsrpcutpu.h
```

Purpose

To configure injection and ignition.

Description

The function initializes the injection and ignition patterns for a specific cylinder. The corresponding parameter set comprises both the injection and the ignition data, and is updated as a whole.

The resolution of injection pulses depends on the TPU prescaler value (TCR1):

Prescaler Value	Resolution [ns]
2	35.7
4	71.4
8	143
14	250
28	500
42	750
56	1000
84	1500
112	2000
168	3000
224	4000

Prescaler Value	Resolution [ns]
336	6000
448	8000

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the update values of the pulse patterns from the master to the slave processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

InjPattern Specifies the initial injection pulse pattern. For the data structure, refer to dsrpcu_tpu_inj_t on page 17.

IgnPattern Specifies the initial ignition pulse pattern. For the data structure, refer to dsrpcu_tpu_ign_t on page 18.

Note

Start angles must always be strictly monotonic increasing. Otherwise injection and ignition pulse generation lead to unpredictable results. Refer to Specification of Injection and Ignition Pulse Patterns.

InjUnit Time-processing unit used for the injection:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

IgnUnit Time-processing unit used for the ignition:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

InjChannel Injection channel number [2 … 14]. An injection channel always uses two TPU channels. You have to specify the first, even one, the second, odd one is chosen automatically. If no injection is desired, use DSRPCU_TPU_NO_INJ_CHANNEL.

IgnChannel Ignition channel number [1 … 16]. If no ignition is desired, use DSRPCU_TPU_NO_IGN_CHANNEL.

Note

The Ch01 channel (channel number 1) of the related TPU is internally used and, thus, not available.

InjPriority Priority of the injection channel. A channel of higher priority is serviced more frequently.

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Note

All engine control functions must have the same priority level, but dsrpcu_tpu_crank_pm_init2 and dsrpcu_tpu_crank_od_init must use channels with a smaller channel number.

IgnPriority Priority of the ignition channel. A channel of higher priority is serviced more frequently.

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Note

All engine control functions must have the same priority level, but dsrpcu_tpu_crank_pm_init2 on page 199 and dsrpcu_tpu_crank_od_init on page 205 must use channels with a smaller channel number.

CylinderNo Number of the cylinder [1 … 12].

TDCAngle Specifies the top dead center angle for the cylinder chosen. Range is [0 … <720].

InjOverlapMode Lets you choose the injection behavior if two or more injection pulses overlap.

- DSRPCU_TPU_INJ_MERGE_PULSES: The pulses are merged, that is, one new pulse is specified:

  Start angle = Start angle of the pulse that started first

  End angle = End angle of the pulse that started last
- DSRPCU_TPU_INJ_REMOVE_PULSES: Only the first pulse is generated, all other pulses are cleared.

Refer to Specification of Injection and Ignition Pulse Patterns (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

IgnOverlapMode Lets you choose the ignition behavior if two or more ignition pulses overlap.

- DSRPCU_TPU_IGN_MERGE_PULSES: The pulses are merged, that is, one new pulse is specified:

  Start angle = Start angle of the pulse that started first

  End angle = Largest end angle
- DSRPCU_TPU_IGN_REMOVE_PULSES: Only the first pulse is generated, all other pulses are cleared.

Refer to Specification of Injection and Ignition Pulse Patterns (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

GateMode Specifies whether the pins for pick-up duration and hold duration are connected via a logical OR.

- DSRPCU_TPU_INJ_OUTPUT_OR_GATED
- DSRPCU_TPU_INJ_OUTPUT_NOT_OR_GATED

InjPolarity Specifies whether a high active or a low active injection signal is to be generated.

- DSRPCU_TPU_INJ_HIGH_ACTIVE
- DSRPCU_TPU_INJ_LOW_ACTIVE

Note

If you configure RapidPro hardware with ConfigurationDesk for RapidPro the actual polarity of a signal results from the interplay of the signal polarity settings of both the RapidPro system and ConfigurationDesk for RapidPro.

IgnPolarity Specifies whether a high active or a low active ignition signal is to be generated.

- DSRPCU_TPU_IGN_HIGH_ACTIVE
- DSRPCU_TPU_IGN_LOW_ACTIVE

Note

If you configure RapidPro hardware with ConfigurationDesk for RapidPro the actual polarity of a signal results from the interplay of the signal polarity settings of both the RapidPro system and ConfigurationDesk for RapidPro.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7900	Error	MSC: memory allocation error.	The memory allocation for internal data storage failed.
0x7903	Error	MSC: dual-port memory allocation error.	The memory allocation for internal data storage failed.
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7FA4	Error	TPU Inj Ign Init: illegal TPU unit number for the inj channels specified.	The specified value is invalid. Use one of the predefined symbols to specify the InjUnit parameter.
0x7FA5	Error	TPU Inj Ign Init: illegal TPU unit number for the ign channels specified.	The specified value is invalid. Use one of the predefined symbols to specify the IgnUnit parameter.
0x7FA6	Error	TPU Inj Ign Init: illegal injection channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the InjChannel parameter.
0x7FA7	Error	TPU Inj Ign Init: illegal ignition channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the IgnChannel parameter.
0x7FA8	Error	TPU Inj Ign Init: inj channel X on TPU Y was reserved by another function.	The specified injection channel is in use. Choose another injection channel for your function.
0x7FA9	Error	TPU Inj Ign Init: ign channel X on TPU Y was reserved by another function.	The specified ignition channel is in use. Choose another ignition channel for your function.
0x7FAA	Error	TPU Inj Ign Init: illegal value for the injection priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the InjPriority parameter.
0x7FAB	Error	TPU Inj Ign Init: illegal value for the ignition priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the IgnPriority parameter.
0x7FAC	Error	TPU Inj Ign Init: illegal injection overlap mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the InjOverlapMode parameter.
0x7FAD	Error	TPU Inj Ign Init: illegal ignition overlap mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the IgnOverlapMode parameter.
0x7FAE	Error	TPU Inj Ign Init: illegal injection gate mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the GateMode parameter.
0x7FAF	Error	TPU Inj Ign Init: illegal value for the injection pulse number specified.	The specified value is invalid. Refer to dsrpcu_tpu_inj_t on page 17.
0x7FB0	Error	TPU Inj Ign Init: illegal value for the ignition pulse number specified.	The specified value is invalid. Refer to dsrpcu_tpu_ign_t on page 18.
0x7FB1	Error	TPU Inj Ign Init: illegal value for the pick up duration specified.	The specified value is invalid. Refer to dsrpcu_tpu_inj_t on page 17.
0x7FB2	Error	TPU Inj Ign Init: illegal value for the X. injection start angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_inj_t on page 17.
0x7FB3	Error	TPU Inj Ign Init: illegal value for the X. injection duration specified.	The specified value is invalid. Refer to dsrpcu_tpu_inj_t on page 17.
0x7FB4	Error	TPU Inj Ign Init: illegal value for the X. ignition start angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_ign_t on page 18.
0x7FB5	Error	TPU Inj Ign Init: illegal value for the X. ignition end angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_ign_t on page 18.
0x7FB6	Error	TPU Inj Ign Init: illegal value for the cylinder number specified.	The specified value is invalid. The specified value is outside the range permitted for the CylinderNo parameter.

ID	Type	Message[1)]	Description
0x7FB7	Error	TPU Inj Ign Init: illegal value for the top dead center angle specified.	The specified value is invalid. The specified value is outside the range permitted for the TDCAngle parameter.
0x7FB8	Error	TPU Inj Ign Init: illegal injection polarity specified.	The specified value is invalid. Use one of the predefined symbols to specify the InjPolarity parameter.
0x7FB9	Error	TPU Inj Ign Init: illegal ignition polarity specified.	The specified value is invalid. Use one of the predefined symbols to specify the IgnPolarity parameter.
0x7FBA	Error	TPU Inj Ign Init: wrong protocol version. Update application firmware.	The use of this function requires a new RPCU application firmware version, refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7FBB	Error	TPU Inj Ign Init: illegal value for the boost duration specified.	The specified value is invalid. Refer to dsrpcu_tpu_inj_t.
0x3F0F	Info	TPU Inj Ign Init: channel X and Y on TPU Z are initialized as inj outputs.	This message is generated if the initialization of injection outputs was successful and the application was compiled using the -DDEBUG_INIT option.
0x3F10	Info	TPU Inj Ign Init: channel X on TPU Y is initialized as ignition output.	This message is generated if the initialization of ignition outputs was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

Basics

Specification of Injection and Ignition Pulse Patterns (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

References

dsrpcu_tpu_inj_ign_start 247
dsrpcu_tpu_inj_ign_stop 247
dsrpcu_tpu_inj_ign_update 245

dsrpcu_tpu_inj_ign_update

Syntax

```
Int16 dsrpcu_tpu_inj_ign_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   dsrpcu_tpu_inj_t* InjPattern,
   dsrpcu_tpu_ign_t* IgnPattern)
```

Include file

```
dsrpcutpu.h
```

Purpose

To update the injection and ignition pulse patterns of a certain cylinder during run-time.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the update values of the pulse patterns from the master to the slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

InjPattern Specifies the injection pulse pattern, refer to dsrpcu_tpu_inj_t on page 17.

IgnPattern Specifies the ignition pulse pattern, refer to dsrpcu_tpu_ign_t on page 18.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_inj_ign_start

Syntax

```
Int16 dsrpcu_tpu_inj_ign_start(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To start the injection and ignition pulse generation for all channels.

Description

As soon as the engine is synchronized, injection and ignition pulse generation is actually started.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_inj_ign_init 240
dsrpcu_tpu_inj_ign_stop 247
dsrpcu_tpu_inj_ign_update 245

dsrpcu_tpu_inj_ign_stop

Syntax

```
Int16 dsrpcu_tpu_inj_ign_stop(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose	To stop injection and ignition pulse generation for all channels.
I/O mapping	None
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.
Return value	**DSRPCU_NO_ERROR** No error occurred. **DSMSC_CMD_BUFFER_OVERFLOW** Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.
Related topics	References dsrpcu_tpu_inj_ign_init ... 240 dsrpcu_tpu_inj_ign_start ... 247 dsrpcu_tpu_inj_ign_update ... 245

Angle-Angle-Based Pulse Generation

Purpose

The RapidPro system uses the following RTLib functions for specifying an angle-angle-based pulse sequence.

Tip

For angle-angle-based pulse generation, it does not matter whether the crankshaft/camshaft signal measurement is set up via TPU- or I/O PLD-related functions.

Where to go from here

Information in this section

dsrpcu_tpu_aabp_init....250
To configure angle-angle based pulse generation.

dsrpcu_tpu_aabp_start....253
To start angle-angle based pulse generation for all channels.

dsrpcu_tpu_aabp_update....254
To update the angle-angle based pulse patterns of a certain cylinder during run-time.

dsrpcu_tpu_aabp_stop....255
To stop angle-angle based pulse generation for all channels.

Information in other sections

Generating Angle-Angle-Based Pulses (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)
To implement the generation of angle-angle-based pulses specified by start and end angles.

dsrpcu_tpu_aabp_init

Syntax

```
void dsrpcu_tpu_aabp_init(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   dsrpcu_tpu_aabp_t* AABPPattern,
   UInt16 AABPUnit,
   UInt16 AABPChannel,
   UInt16 AABPPriority,
   UInt16 AABPIndex,
   Float64 TDCAngle,
   UInt16 AABPOverlapMode,
   UInt16 AABPPolarity)
```

Include file

```
dsrpcutpu.h
```

Purpose

To configure angle-angle based pulse generation.

Description

The function initializes an angle-angle based pulse pattern.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the update values of the pulse patterns from the master to the slave processor. The parameter set is

assigned to a specific channel/unit and initialized when the function is being executed.

AABPPattern Specifies the initial angle-angle based pulse pattern. For the data structure, refer to dsrpcu_tpu_aabp_t on page 19.

Note

Start angles must always be strictly monotonic increasing. Otherwise pulse generation lead to unpredictable results.

AABPUnit Time-processing unit used for the angle-angle based pulse generation:

- DSRPCU_TPU_A
- DSRPCU_TPU_B
- DSRPCU_TPU_C

AABPChannel Channel number [1 … 16].

Note

The Ch01 channel (channel number 1) of the related TPU is internally used and, thus, not available.

AABPPriority Priority of the angle-angle based pulse channel. A channel of higher priority is serviced more frequently.

- DSRPCU_TPU_CH_PRIOR_LOW
- DSRPCU_TPU_CH_PRIOR_MID
- DSRPCU_TPU_CH_PRIOR_HIGH

Note

All engine control functions must have the same priority level, but dsrpcu_tpu_crank_pm_init2 and dsrpcu_tpu_crank_od_init must use channels with a smaller channel number.

AABPIndex Used for internal purposes only. You do not need to specify this parameter. Initialized to 1.

TDCAngle Specifies the top dead center angle for the cylinder chosen. Range is [0 … <720].

AABPOverlapMode Lets you choose the ignition behavior if two or more ignition pulses overlap.

- DSRPCU_TPU_AABP_MERGE_PULSES: The pulses are merged, that is, one new pulse is specified:

  Start angle = Start angle of the pulse that started first

  End angle = Largest end angle
- DSRPCU_TPU_AABP_REMOVE_PULSES: Only the first pulse is generated, all other pulses are cleared.

Same "Remove" behavior as for ignition pulses, refer to Specification of Injection and Ignition Pulse Patterns (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

AABPPolarity Specifies whether a high active or a low active ignition signal is to be generated.

- DSRPCU_TPU_AABP_HIGH_ACTIVE
- DSRPCU_TPU_AABP_LOW_ACTIVE

Note

If you configure RapidPro hardware with ConfigurationDesk for RapidPro the actual polarity of a signal results from the interplay of the signal polarity settings of both the RapidPro system and ConfigurationDesk for RapidPro.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7900	Error	MSC: memory allocation error.	The memory allocation for internal data storage failed.
0x7903	Error	MSC: dual port memory allocation error	The memory allocation for internal data storage failed.
0x7F06	Error	TPU: memory allocation error.	The memory allocation for internal data storage failed.
0x7FD0	Error	TPU AABP Init: wrong protocol version. Update application firmware	The use of this functions requires a new RPCU application firmware version (1.4 or later), refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7FD1	Error	TPU AABP Init: illegal channel number specified.	The specified value is invalid. The specified value is outside the range permitted for the AABPChannel parameter.
0x7FD2	Error	TPU AABP Init: it is not allowed to initialize more than 12 AABP channels	The specified value is invalid. It is not allowed to initialize more than 12 AABP channels.
0x7FD3	Error	TPU AABP Init: illegal value for the top dead center angle specified.	The specified value is invalid. The specified value is outside the range permitted for the TDCAngle parameter.
0x7FD4	Error	TPU AABP Init: illegal TPU unit number for the aabp channels specified.	The specified value is invalid. Use one of the predefined symbols to specify the AABPUnit parameter.
0x7FD5	Error	TPU AABP Init: channel X on TPU Y was reserved by another function.	The specified ignition channel is in use. Choose another ignition channel for your function.
0x7FD6	Error	TPU AABP Init: illegal value for the priority specified.	The specified value is invalid. Use one of the predefined symbols to specify the AABPPriority parameter.
0x7FD7	Error	TPU AABP Init: illegal AABP overlap mode parameter specified.	The specified value is invalid. Use one of the predefined symbols to specify the AABPOverlapMode parameter.
0x7FD8	Error	TPU AABP Init: illegal AABP polarity specified.	The specified value is invalid. Use one of the predefined symbols to specify the AABPPolarity parameter.

ID	Type	Message[1)]	Description
0x7FD9	Error	TPU AABP Init: illegal value for the AABP pulse number specified.	The specified value is invalid. Refer to dsrpcu_tpu_aabp_t on page 19.
0x7FDA	Error	TPU AABP Init: illegal value for the X. AABP start angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_aabp_t on page 19.
0x7FDB	Error	TPU AABP Init: illegal value for the X. AABP end angle specified.	The specified value is invalid. Refer to dsrpcu_tpu_aabp_t on page 19.
0x3F15	Info	TPU AABP Init: channel X on TPU Y is initialized as AABP.	This message is generated if the initialization of AABP was successful and the application was compiled using the -DDEBUG_INIT option.

[1)] The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_tpu_aabp_start..... 253
dsrpcu_tpu_aabp_stop..... 255
dsrpcu_tpu_aabp_update..... 254

dsrpcu_tpu_aabp_start

Syntax

```
Int16 dsrpcu_tpu_aabp_start(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To start angle-angle based pulse generation for all channels.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_aabp_init..... 250
dsrpcu_tpu_aabp_stop..... 255

dsrpcu_tpu_aabp_update

Syntax

```
Int16 dsrpcu_tpu_aabp_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   dsrpcu_tpu_aabp_t* AABPPattern)
```

Include file

```
dsrpcutpu.h
```

Purpose

To update the angle-angle based pulse patterns of a certain cylinder during run-time.

I/O mapping

The I/O mapping between the specified signals and the I/O connector pins depends on the installed modules and the connected RapidPro units. The I/O mapping information is specific to your RapidPro system and can be displayed in ConfigurationDesk for RapidPro. For further information, refer to Signal Mapping to I/O Pins (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Start address of the allocated parameter set. The parameter set is used to transfer the update values of the pulse patterns from the master to the

slave processor. The parameter set is allocated and initialized by the associated initialization function (`dsrpcu_[…]_init`).

AABPPattern Specifies the new angle-angle based pulse pattern. For the data structure, refer to dsrpcu_tpu_aabp_t on page 19.

Note

Start angles must always be strictly monotonic increasing. Otherwise pulse generation lead to unpredictable results.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_PARAM_SET_ACCESS_ERROR Master-slave communication error. The update values have not been sent to the slave.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_aabp_init..... 250
dsrpcu_tpu_aabp_start..... 253
dsrpcu_tpu_aabp_stop..... 255

dsrpcu_tpu_aabp_stop

Syntax

```
Int16 dsrpcu_tpu_aabp_stop(
   dsrpcu_access_t* AccessPtr)
```

Include file

```
dsrpcutpu.h
```

Purpose

To stop angle-angle based pulse generation for all channels.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_tpu_aabp_init 250
dsrpcu_tpu_aabp_start 253
dsrpcu_tpu_aabp_update 254

Diagnostic

Where to go from here

Information in this section

Structure of the Diagnostic Data Array

Objective

Each module that provides diagnostic information stores it in a diagnostic data array. You must allocate the required size of the data array and you must know the position of the diagnostic information within this array to be read.

Diagnostic data array

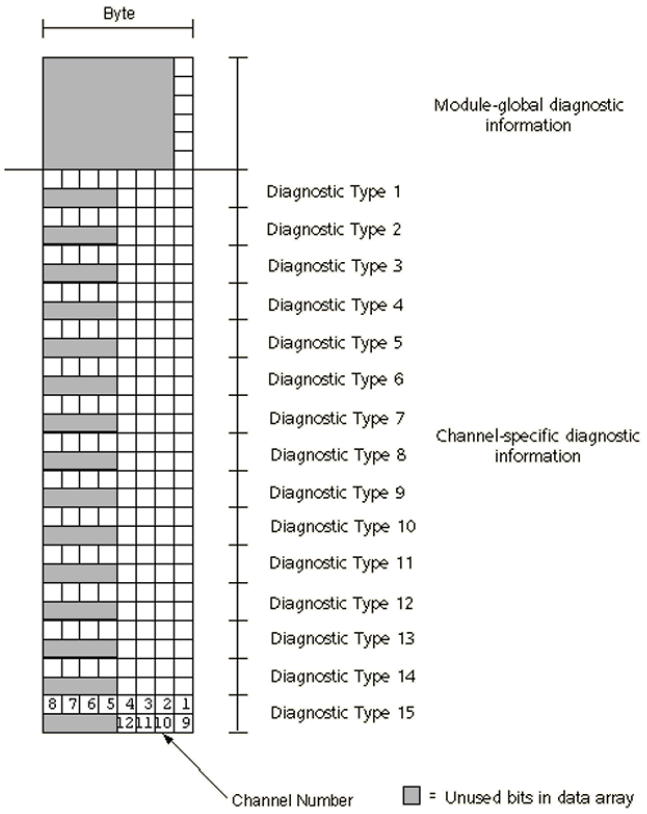
The diagnostic data array contains both the module-global data and the channel-specific data:

- Module-global diagnostic information

  The relevant bit in each byte (LSB) contains the diagnostic information for one diagnostic type which is offered by the module.
- Channel-specific diagnostic information

  The number of bytes varies for each diagnostic type depending on the number of channels for which diagnostic information must be transferred. Each bit in the bytes represents the diagnostic information for one channel. One byte contains the data for up to 8 channels.

The following figure shows the structure of the diagnostic data array.



Allocating memory for the diagnostic data array

If you want to read diagnostic information from a module, you must allocate a data array beforehand. You can calculate the required memory by the following formula:

$Size_{DataArray} = Size_{Module} + Size_{Channel} \cdot n_{Byte}$

with:

$Size_{Module}$: Number of available module-global diagnostic types

$Size_{Channel}$: Number of available channel-specific diagnostic types

n_{Byte}: Number of Bytes to be allocated, depends on the number of channels (1 … 8 channels require 1 Byte, 9 … 16 channels require 2 Bytes, …)

Note

The size of the data array must always an even number. If the calculated data array size is an odd number, you must add 1 to it.

Getting the available diagnostic types

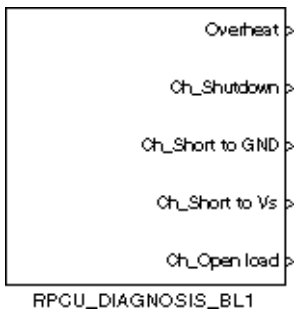
If you want to calculate the size of the diagnostic data array, you must know the number of module-global and channel-specific diagnostic types of the module you want to get the diagnostic information from. To get this information, there are two methods:

- Look at the module's data sheet in the *RapidPro System Installation and Configuration Reference*, for example PS-LSD 6/1 Data Sheet (cut-out).

Parameter	Specification
…	…
Diagnostics	Open load, short to GND, short to supply, overheat, shutdown
…	…

  The PS-LSD 6/1 module supports five diagnostic types: Open load, short to GND, short to supply, overheat, and shutdown.

- Open a Simulink model, drag an RPCU_DIAGNOSIS_BLx block to it and make the corresponding settings for the module you want to read the diagnostic information from. After parameterizing the block, there are outports for each available diagnostic type. Channel-specific diagnostic types start with a "Ch_".



Processing a diagnostic request

To get the diagnostic information, you must use the following RTLib functions for each module you want to get the diagnostic information from:

1. `dsrpcu_diag_init`
2. `dsrpcu_diag_request`
3. `dsrpcu_diag_read`

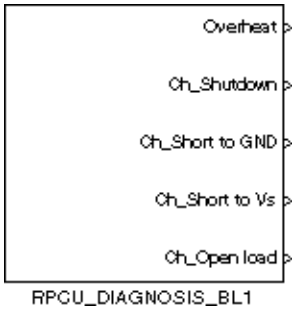
   The read function contains the `DiagData` parameter. The calculated buffer size must be allocated, for example:

```
UInt8 DiagData[SIZE_DATA_ARRAY];
…
dsrpcu_diag_read (MyAccessPtr, MyParamSetPtr,
   (UInt16 *)DiagData, &Status);
```

Example

The following example shows the structure of the diagnostic data array for a PS-LSD 6/1 module.

The appropriate RPCU_DIAGNOSIS_BLx block looks like this:



The module has got 1 module-global diagnostic type and 4 channel-specific diagnostic types. It provides 6 channels.

The size of the diagnostic data array to be allocated is:

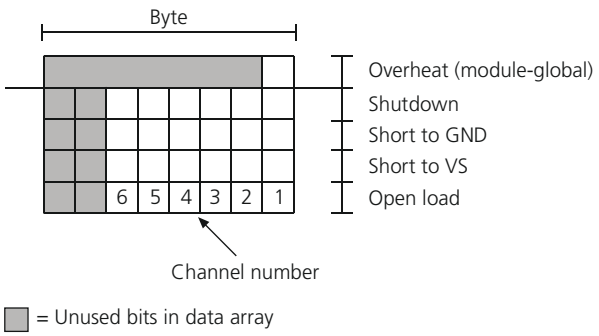
$Size_{DataArray} = Size_{Module} + Size_{Channel} \cdot n_{Byte}$

$Size_{DataArray} = 1 + 4 \cdot 1 = 5$

The data array size must be even. You have to add 1.

$Size_{DataArray} = 6$

The corresponding structure of the diagnostic data array is:



dsrpcu_diag_alive_init

Syntax

```
void dsrpcu_diag_alive_init (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 BoardType,
   UInt16 LayerNo)
```

Include file

DsRPCUDiag.h

Purpose

To initialize the monitoring of the alive state for one carrier board.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer and request the diagnosis alive status for one carrier board from the slave to the master processor. The parameter set is assigned to a specific carrier board and initialized when the function is being executed.

BoardType Specifies the board type:

- DSRP_DIAG_RPCU_BOARD_ID: Control Unit
- DSRP_DIAG_RPPU_BOARD_ID: Power Unit
- DSRP_DIAG_RPSCU_BOARD_ID: Signal Conditioning Unit

LayerNo Number of the layer in the stack where the carrier board is located. Range is [1 … 15]. 1 means bottom.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7520	Error	Diag Alive Init: illegal board type specified.	The specified value is invalid. Use one of the predefined symbols to specify the BoardType parameter.
0x7521	Error	Diag Alive Init: illegal layer number specified.	The specified value is invalid. The specified value is outside the range permitted for the LayerNo parameter.

ID	Type	Message[1)]	Description
0x3501	Info	Diag Alive Init: unit %d initialized for diagnosis.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_diag_alive_read..... 263
dsrpcu_diag_alive_request..... 262

dsrpcu_diag_alive_request

Syntax

```
Int16 dsrpcu_diag_alive_request (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

DsRPCUDiag.h

Purpose

To request the current diagnosis alive status for a certain carrier board.

Description

The following alive statuses are possible:

- DSRPCU_DIAG_UNIT_ALIVE
- DSRPCU_DIAG_UNIT_NOT_ALIVE

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to request and transfer the diagnosis alive counter from the slave to the master processor. The parameter set is assigned to a specific carrier board and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

DSRPCU_NO_ERROR: No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW: Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_diag_alive_init 261
dsrpcu_diag_alive_read 263

dsrpcu_diag_alive_read

Syntax

```
Int16 dsrpcu_diag_alive_read (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt16* AliveStatus,
   UInt16* Status)
```

Include file

DsRPCUDiag.h

Purpose

To read the most recent alive status of a certain carrier board.

Description

This function reads the most recent alive status of a certain carrier board which was requested by the **`dsrpcu_diag_alive_read()`** function, refer to dsrpcu_diag_alive_request on page 262.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the **`dsrpcu_init()`** function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to transfer the diagnosis data from the slave to the master processor. The

parameter set is assigned to a specific carrier board and initialized by the associated initialization function (`dsrpcu_[…]_init`).

AliveStatus Address where the current alive status of the carrier board is stored.

- DSRPCU_DIAG_UNIT_ALIVE
- DSRPCU_DIAG_UNIT_NOT_ALIVE

Status Address where the current status of the alive-status information is stored:

- DSRPCU_NEW_VALUE
- DSRPCU_OLD_VALUE

Return value

DSRPCU_NO_ERROR: No error occurred.

Related topics

References

dsrpcu_diag_alive_init 261
dsrpcu_diag_alive_request 262

dsrpcu_diag_init

Syntax

```
void dsrpcu_diag_init (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 BoardType,
   UInt16 LayerNo,
   UInt16 SlotNumOf,
   UInt16* SlotNo,
   Uint16* SlotChnlNo,
   UInt16 ModuleType,
   UInt16 ModuleVS,
   UInt16 ModDiagSize,
   UInt16 ChnlDiagSize)
```

Include file

DsRPCUDiag.h

Purpose

To initialize the diagnosis for one module on one carrier board.

Note

A maximum of 47 modules can be initialized for diagnosis.

I/O mapping None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the data for one module from the slave to the master processor. The parameter set is assigned to a specific module and initialized when the function is being executed.

BoardType Specifies the board type that contains the module and for which diagnosis has to be performed:

- DSRP_DIAG_RPCU_BOARD_ID: Control Unit
- DSRP_DIAG_RPPU_BOARD_ID: Power Unit
- DSRP_DIAG_RPSCU_BOARD_ID: Signal Conditioning Unit

LayerNo Number of the layer in the stack where the carrier board is located. Range is [1 … 15]. 1 means bottom (RapidPro Control Unit).

SlotNumOf Total number of slots the module is connected to.

SlotNo Pointer to an array that holds the ID number of the slots which the module is connected to. The total number of array elements is specified by SlotNumOf.

SlotChnlNo Pointer to an array that holds the number of channels of each slot. The total number of array elements is specified by SlotNumOf.

Note

For detailed information on the RapidPro hardware modules, refer to the RapidPro System Hardware Reference 🕮 document.

ModuleType DS number of the module for which diagnostic data is requested, for example, 1661 (DS1661 board).

Note

The ModuleType parameter is not supported in this version.

ModuleVS Hardware version of the module for which diagnostic data is requested, for example, 4 (DS1661-04).

Note

The ModuleVS parameter is not supported in this version.

ModDiagSize Total number of the possible global diagnostic messages, for example, 1 for the DS1633, as one global diagnostic message can occur: Shutdown.

ChnlDiagSize Total number of the possible channel-specific diagnostic messages, for example, 0 for the DS1633, as no channel-specific diagnostic messages can occur.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7500	Error	DIAG: memory allocation error.	The memory allocation for internal data storage failed.
0x7510	Error	DIAG Init: maximum number of diagnosis modules exceeded.	Too many modules have been initialized. Do not initialize more than 47 modules for diagnosis.
0x7511	Error	Diag Init: illegal board type specified.	The specified value is invalid. Use one of the predefined symbols to specify the BoardType parameter.
0x7512	Error	Diag Init: illegal layer number specified.	The specified value is invalid. The specified value is outside the range permitted for the LayerNo parameter.
0x7513	Error	Diag Init: illegal number of module slots specified.	The specified value is invalid. Use one of the predefined symbols to specify the SlotNumOf parameter.
0x7514	Error	Diag Init: illegal slot number specified.	The specified value is invalid. Use one of the predefined symbols to specify the SlotNo parameter.
0x7515	Error	Diag Init: illegal number of slot channel specified.	The specified value is invalid. Use one of the predefined symbols to specify the SlotChnlNo parameter.
0x3500	Info	Diag Init: module on slot %d on unit %d initialized for diagnosis.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

Basics

Structure of the Diagnostic Data Array 258

References

dsrpcu_diag_read 267
dsrpcu_diag_request 267
RapidPro System Hardware Reference

dsrpcu_diag_request

Syntax

```
Int16 dsrpcu_diag_request (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

DsRPCUDiag.h

Purpose

To request current diagnosis data for one particular module.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to request and transfer the diagnosis data from the slave to the master processor. The parameter set is assigned to a specific carrier board and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

- DSRPCU_NO_ERROR: No error occurred.
- DSMSC_CMD_BUFFER_OVERFLOW: Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_diag_init 264
dsrpcu_diag_read 267

dsrpcu_diag_read

Syntax

```
Int16 dsrpcu_diag_read (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   UInt16* DiagData,
   UInt16* Status)
```

Include file	DsRPCUDiag.h
Purpose	To read diagnosis data requested by the dsrpcu_diag_request function.
Description	To read the diagnosis data that was requested by the dsrpcu_diag_request function. You must interpret the data with respect to the module type. For detailed information on the RapidPro hardware modules, refer to the RapidPro System Hardware Reference 🕮 document.
I/O mapping	None
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **ParamSetPtr** Address of the allocated parameter set. The parameter set is used to transfer the diagnosis data from the slave to the master processor. The parameter set is assigned to a specific carrier board and initialized by the associated initialization function (`dsrpcu_[…]_init`). **DiagData** Start address of an array where the diagnosis data read is stored. You must maintain that enough memory is available. For detailed information on the diagnosis data, refer to Diagnostic Information Provided by the RapidPro System (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮). **Status** Address where the current status of the diagnosis data access is stored: ▪ DSRPCU_NEW_VALUE ▪ DSRPCU_OLD_VALUE
Return value	**DSRPCU_NO_ERROR** No error occurred.
Related topics	Basics Diagnostic Information Provided by the RapidPro System (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮) Structure of the Diagnostic Data Array ... 258 References dsrpcu_diag_init ... 264 dsrpcu_diag_request ... 267

dsrpcu_diag_module_reset_init

Syntax

```
void dsrpcu_diag_module_reset_init (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 BoardType,
   UInt16 LayerNo,
   UInt16 SlotNumOf,
   UInt16* FirstSlotNo)
```

Include file

DsRPCUDiag.h

Purpose

To initialize a module for a diagnosis data reset.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to request and transfer the reset information from the slave to the master processor.

BoardType Specifies the board type that contains the module and for which diagnosis has to be performed:

- DSRP_DIAG_RPCU_BOARD_ID: Control Unit
- DSRP_DIAG_RPPU_BOARD_ID: Power Unit
- DSRP_DIAG_RPSCU_BOARD_ID: Signal Conditioning Unit

LayerNo Number of the layer in the stack where the carrier board is located. Range is [1 … 15]. 1 means bottom (RapidPro Control Unit).

SlotNumOf Total number of slots the module is connected to.

FirstSlotNo Number of the first slot the module is connected to.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7500	Error	DIAG: memory allocation error.	The memory allocation for internal data storage failed.
0x7540	Error	Diag Module Reset Init: wrong protocol version. Please update appl. firmware.	The use of this function requires a new RPCU application firmware version (1.4 or later), refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x7541	Error	Diag Module Reset Init: maximum number of diagnosis modules exceeded.	Total number of modules diagnosis data reset has to be performed to is exceeded.
0x7542	Error	Diag Module Reset Init: illegal board type specified.	The specified value is invalid. Use one of the predefined symbols to specify the BoardType parameter.
0x7543	Error	Diag Module Reset Init: illegal layer number specified.	The specified value is outside the range permitted for the LayerNo parameter.
0x7544	Error	Diag Module Reset Init: illegal number of module slots specified.	The specified value is outside the range permitted for the SlotNumOf parameter.
0x7545	Error	Diag Module Reset Init: illegal first slot number specified.	The specified value is outside the range permitted for the FirstSlotNo parameter.
0x3503	Info	Diag Module Reset Init: module on slot %d on unit %d initialized for reset.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_diag_module_reset..... 270

dsrpcu_diag_module_reset

Syntax

```
Int16 dsrpcu_diag_module_reset (
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr)
```

Include file

DsRPCUDiag.h

Purpose

To perform a module-specific reset of diagnosis data.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Address of the allocated parameter set. The parameter set is used to request and transfer the reset information from the slave to the master processor. The parameter set is assigned to a specific carrier board and initialized by the associated initialization function (`dsrpcu_[…]_init`).

Return value

- DSRPCU_NO_ERROR: No error occurred.
- DSMSC_CMD_BUFFER_OVERFLOW: Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_diag_module_reset_init..... 269

dsrpcu_diag_global_reset_init

Syntax

```
void dsrpcu_diag_module_reset_init (
   dsrpcu_access_t* AccessPtr)
```

Include file

DsRPCUDiag.h

Purpose

To initialize all modules for a global diagnosis data reset.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7500	Error	DIAG: memory allocation error.	The memory allocation for internal data storage failed.
0x7530	Error	Diag Global Reset Init: wrong protocol version. Please update appl. firmware.	The use of this function requires a new RPCU application firmware version (1.4 or later), refer to How to Update RapidPro Firmware (RapidPro System Hardware Installation Guide 🕮).
0x3502	Info	Diag Global Reset Init: module on slot %d on unit %d initialized for reset.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_diag_global_reset 272

dsrpcu_diag_global_reset

Syntax

```
Int16 dsrpcu_diag_global_reset (
   dsrpcu_access_t* AccessPtr)
```

Include file

DsRPCUDiag.h

Purpose

To perform a global reset of diagnosis data for all modules.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

- DSRPCU_NO_ERROR: No error occurred.
- DSMSC_CMD_BUFFER_OVERFLOW: Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_diag_module_reset_init 269

Interrupt Handling

General note

Note

The interrupt handling of RapidPro is bit-based and replaces the standard word-based interrupt handling of the ECU interface.

For basic information on interrupt handling, refer to Interrupts Provided by the RapidPro Control Unit (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮).

Objective

To make use of angle-based interrupts.

Where to go from here

Information in this section

Information in other sections

Interrupts Provided by the RapidPro Control Unit (RapidPro System – I/O Subsystem MPC565 Implementation Features 🕮)

dsrpcu_per_angle_int_init

Syntax

```
void dsrpcu_per_angle_int_init(
   dsrpcu_access_t* AccessPtr,
   UInt16 PerIntIndex,
   Float64 StartAngle,
   Float64 PeriodAngle)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To initialize angle-based interrupt generation for periodic interrupts.

Note

The dsrpcu_tpu_crank_pm_init2 function and the dsrpcu_tpu_cam_init3 function have to be called first.

Description

The first interrupt is triggered at a defined angle. Further interrupts are triggered periodically. Up to six periodic interrupts can be defined, each of which requires its own function call.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

PerIntIndex ID of the periodic interrupt. Range is [1 … 6].

StartAngle Start angle of periodic interrupt generation. Range is [0° … <720°].

PeriodAngle Angle interval of periodic interrupt generation. Range is [0.1° … 720°].

Note

The PeriodAngle parameter cannot be modified during run-time.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7701	Error	ADDIO: memory allocation error.	The memory allocation for internal data storage failed.
0x7710	Error	ADDIO Period Angle Int Init: illegal angle period index specified.	The specified value is invalid. The specified value is outside the range permitted for the PerIntIndex parameter.
0x7711	Error	ADDIO Period Angle Int Init: illegal value for the start angle specified.	The specified value is invalid. The specified value is outside the range permitted for the StartAngle parameter.
0x7712	Error	ADDIO Period Angle Int Init: illegal value for the period angle specified.	The specified value is invalid. The specified value is outside the range permitted for the PeriodAngle parameter.
0x3700	Info	ADDIO Period Angle Int Init: angle period sub-interrupt %d initialized.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. “DS1007: M1”, “DS1401: (0x01IX)”.

Related topics

References

dsrpcu_angle_int_init..... 277
dsrpcu_angle_int_update..... 278

dsrpcu_angle_int_init

Syntax

```
void dsrpcu_angle_int_register(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t** ParamSetPtr,
   UInt16 IntIndex,
   Float64 Angle)
```

Include file

```
dsrpcuaddio.h
```

Purpose

To initialize a subinterrupt which is triggered at a defined engine angle.

Description

Up to 16 interrupts can be defined, each of which requires its own function call.

I/O mapping

None

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

ParamSetPtr Parameter that holds the start address of the allocated parameter set. The parameter set is used to transfer the new interrupt angle values from the master to the slave processor. The parameter set is assigned to a specific channel/unit and initialized when the function is being executed.

IntIndex Index of the used sub-interrupt. Range is [1 … 16].

Angle Position of the subinterrupt. Range is [0° … <720°].

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7713	Error	ADDIO Angle Int Init: illegal value for the interrupt number specified.	The specified value is invalid. The specified value is outside the range permitted for the IntIndex parameter.
0x7714	Error	ADDIO Angle Int Init: illegal value for the angle specified.	The specified value is invalid. The specified value is outside the range permitted for the Angle parameter.
0x3701	Info	ADDIO Angle Int Init: angle sub-interrupt %d initialized.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_angle_int_update..... 278
dsrpcu_per_angle_int_init..... 276

dsrpcu_angle_int_update

Syntax

```
Int16 dsrpcu_angle_int_update(
   dsrpcu_access_t* AccessPtr,
   dsrpcu_param_t* ParamSetPtr,
   Float64 Angle)
```

Include file	`dsrpcuaddio.h`
Purpose	To update the angle position for the subinterrupt generation during run-time.
Description	This function updates the angle position at which a subinterrupt is triggered.
I/O mapping	None
Parameters	**AccessPtr** Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function. **ParamSetPtr** Address of the allocated parameter set. The parameter set is used to transfer data from the master to the slave processor. The parameter set is assigned to a specific channel/unit and initialized by the associated initialization function (`dsrpcu_[…]_init`). **Angle** Position of the subinterrupt. Range is [0° … <720°].
Return value	**DSRPCU_NO_ERROR** No error occurred. **DSMSC_PARAM_SET_ACCESS_ERROR** Master-slave communication error. The update values have not been sent to the slave. **DSMSC_CMD_BUFFER_OVERFLOW** Master-slave communication error. Not enough space in the command buffer to store the command. The command is not sent to the slave.
Related topics	References dsrpcu_angle_int_init ..... 277 dsrpcu_per_angle_int_init 276

System

Where to go from here

Information in this section

Power Supply Management

Where to go from here

Information in this section

dsrpcu_sleep_init

Syntax

```
void dsrpcu_sleep_init (
   dsrpcu_access_t* AccessPtr,
   UInt16 LowPowerMode,
   UInt16 IOPUnitMode)
```

Include file

```
DSRPCUInit.h
```

Purpose

To initialize the sleep mode functions of the RapidPro Control Unit.

Description

The function initializes the sleep mode functions of the RapidPro Control Unit. After successful initialization, the sleep mode can be activated and deactivated via the dsrpcu_sleep and dsrpcu_wake_up functions.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

LowPowerMode Specifies the low-power mode of the MPC565 microprocessor when the sleep mode is activated:

- DSRPCU_LOW_POWER_DOZE
- DSRPCU_LOW_POWER_SLEEP
- DSRPCU_LOW_POWER_DEEPSLEEP

SLEEP mode has a lower power consumption than DOZE, and DEEPSLEEP has the lowest.

IOPUnitMode Specifies whether the Power and SC Units themselves are shut down, in addition to their outputs being disabled, when the sleep mode is active.

- DSRPCU_UNIT_SHUT_DOWN: The entire power supply is switched off.
- DSRPCU_UNIT_NO_SHUT_DOWN

If no Power and SC Units are connected to the RapidPro Control Unit, this parameter is not evaluated.

The outputs of all modules in the RapidPro stack are disabled when the sleep mode is active, independent on the setting of the IOPUnitMode parameter.

Return value

None

Messages

The following messages are defined:

ID	Type	Message[1)]	Description
0x7841	Error	Sleep Init: sleep mode functions have already been initialized.	You can initialize the sleep mode only once.
0x7842	Error	Sleep Init: memory allocation error.	The memory allocation for internal data storage failed.
0x3804	Info	Sleep Init: Sleep mode functions initialized successfully.	This message is generated if the initialization was successful and the application was compiled using the -DDEBUG_INIT option.

1) The message string contains the name of the master used and the module/board number, e.g. "DS1007: M1", "DS1401: (0x01IX)".

Related topics

References

dsrpcu_sleep 283
dsrpcu_wake_up 284

dsrpcu_sleep

Syntax

```
void dsrpcu_sleep (
   dsrpcu_access_t* AccessPtr)
```

Include file

```
DSRPCUInit.h
```

Purpose

To activate the sleep mode of a RapidPro Control Unit.

Description

Software execution is stopped and the outputs of all units in the RapidPro stack are disabled (refer to IOPUnitMode on page 283) until the dsrpcu_wake_up on page 284 function is called.

Note

If the RapidPro Control Unit is already in sleep mode, this function does not have any effect.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command. The command is not sent to the slave.

dsrpcu_wake_up

Syntax

```
void dsrpcu_wake_up (
   dsrpcu_access_t* AccessPtr)
```

Include file

```
DSRPCUInit.h
```

Purpose

To wake up the RapidPro Control Unit from the sleep mode.

Description

Software execution is continued and the outputs of all the Power and SC Units in the RapidPro stack are enabled.

For waking up from DOZE and SLEEP mode, 3-4 clock cycles of the internal MPC565 clock (56 MHz) are necessary. For waking up from DEEPSLEEP mode, <500 clock cycles of the external clock (4 MHz) are necessary. Refer to LowPowerMode on page 282.

Note

If the RapidPro Control Unit is not in sleep mode, this function does not have any effect.

Parameters

AccessPtr Start address of a handle to access the dual-port memory (DPMEM) of the RapidPro hardware. AccessPtr is related to a specific ECU interface channel. The handle is allocated and initialized by the `dsrpcu_init()` function.

Return value

DSRPCU_NO_ERROR No error occurred.

DSMSC_CMD_BUFFER_OVERFLOW Not enough space in the command buffer to store the command. The command is not sent to the slave.

Related topics

References

dsrpcu_sleep 283
dsrpcu_sleep_init 282

C

D

E

L

M

P

S

W