Common Registers and Operation (Device) 1.1

Document Version 1.6

1. Introduction

In order to unify the identification of a Harp Device in a system and to create an environment where users can quickly use a Harp Device, mandatory registers and operation modes should be present. Both must comply with the functionalities described on this document. Note that some of the registers and functionalities are not mandatory.

Common Registers:

Summary description of each register can be found on Table 2-1.

The addresses of the Application Registers should be equal to 32 or higher. These registers differ from device to device according to the device purpose and is up to the developer the names and types of each register.

Operation Modes:

The Harp Device, when acting as a Peripheral, should implement the next Operation Modes:

- **Standby Mode:** Replies to Controller commands. Events are disabled and must not be sent.
- **Active Mode:** Replies to Controller commands. Events are enabled and sent to Controller whenever the Peripheral decides.
- **Speed Mode:** Allows the implementation of a different and specific communication protocol. On this mode, the Harp Binary Protocol is no longer used. The specific protocol designed must implement the possibility to leave from this mode.

The mandatory Operation Modes are the **Standby Mode** and **Active Mode**. The **Speed Mode** is optional and, in many of the applications, is not needed.

It's strongly recommended that a Harp Device acting as Peripehral should continuously check if the communication with the Controller is active and healthy. If this doesn't happen over the last 3 seconds, the Harp Device should go to Standby Mode and flush/destroy its TX buffer.

2. Registers

2.1 Common Registers

Table 2-1. List of available Common Registers

Name	Volatile	Read Only	Type	Add.	Default	Brief Description	Mandatory
R_WHO_AM_I	-	Yes	U16	000	a)	Who am I	Yes
R_HW_VERSION_H	-	Yes	U8	001	a)	Major Hardware version	Yes
R_HW_VERSION_L	-	Yes	U8	002	a)	Minor Hardware version	Yes
R_ASSEMBLY_VERSION	-	Yes	U8	003	a)	Version of the assembled components	Optional
R_HARP_VERSION_H	-	Yes	U8	004	a)	Major HARP version	Optional
R_HARP_VERSION_L	-	Yes	U8	005	a)	Minor HARP version	Optional
R_FW_VERSION_H	-	Yes	U8	006	a)	Major Firmware version of the application	Yes
R_FW_VERSION_L	-	Yes	U8	007	a)	Minor Firmware version of the application	Yes
R_TIMESTAMP_SECOND	Yes	No	U32	008	0	System timestamp: seconds	Yes
R_TIMESTAMP_MICRO	Yes	Yes	U16	009	0	System timestamp: microseconds	Optional
R_OPERATION_CTRL	No	No	U8	010	b)	Configuration of the operation mode	c)
R_RESET_DEV	No	No	U8	011	b)	Reset device and save non-volatile registers	Optional
R_DEVICE_NAME	No	No	U8	012	b)	Name of the device given by the user	Optional
R_SERIAL_NUMBER	No	No	U16	013	b)	Unique serial number of the device	Optional
R_CLOCK_CONFIG	No	No	U8	014	b)	Synchronization clock configuration	Optional

a) These values are stored during factory process and are persistent, i.e., they cannot be changed by the user.
b) Check register notes on the specific register explanation
c) Only parts of the functionality is mandatory. Check register notes on the explanation.

2.1.1 R_WHO_AM_I – Who Am I

Add: 000

WHO_AM_I [15:0]

Used to verify the identity of the device.

A list of devices can be found at https://github.com/harp-tech/protocol. To reserve a range or certain IDs for your project or company, please send an e-mail to filipe@oeps.tech.

If the device doesn't have a pre-allocated ID on the IDs list, this register should be read as 0.

2.1.2 R_TIMESTAMP_SECOND – System timestamp: seconds

Add: 008

SECONDS [31:0]

Contains the current system timestamp in seconds. The default value is 0 (ZERO) and will increment one unit for each second.

2.1.3 R_TIMESTAMP_MICRO – System timestamp: microseconds

Add: 009

USECONDS [15:0]

Contains the microseconds count within each second. Each LSB corresponds to 32 µseconds. The maximum value is 31249.

2.1.4 R_OPERATION_CTRL – Configuration the operation mode

Bit	7	6	5	4	3	2	1	0
Add: 010	ALIVE_EN	OPLEDEN	VISUALEN	MUTE_RPL	DUMP	-	OP_MODE [1:0]	
Default	1	1	1	0	0	0	0	
Mandatory	Optional	Optional	Optional	Optional	Optional	-	a)	

a) Standby Mode and Active Mode are mandatory. Speed Mode is optional.

- **Bits 1:0 – OP_MODE [1:0]: Operation Mode**

These bits define the operation mode of the device.

Note that, if Speed Mode is selected, the device will no longer reply to the HARP commands, only to its specific Speed Mode commands.

- **Bit 3 – DUMP**

When written to 1, the device adds the content of all registers to the streaming buffer.

This bit is always read as 0.

- **Bit 4 – MUTE_RPL**

If equals to 1, the Replies to all the Commands are muted, i.e., will not be sent by the device.

- **Bit 5 – VISUALEN**

If equals to 1, the visual indications, typically LEDs, available on the device will operate. If equals to 0, all the visual indications should turn off.

- **Bit 6 – OPLEDEN**

If equal to 1, the LED present on the device will indicate the Operation Mode selected.

Table 2-2. Operation Mode

OP_MODE [1:0]	Configuration
0	Standby Mode. The device has all the Events turned off.
1	Active Mode. The device turn ON the Events detection. Only the enabled Events will be operating.
2	Reserved.
3	Speed Mode. The device enters into Speed Mode.

Table 2-3. LED toggle indication

Interval in seconds	Operation Mode
4	Standby Mode.
2	Active Mode.
1	Speed Mode.
0.1	A critical error occurred. Only a hardware reset or a new power up can remove the device from this Mode.

- **Bit 7 – ALIVE_EN**

If equal to 1, the device sends an Event with the R_TIMESTAMP_SECONDS content each second. This allows the Controller and/or the user to check that the device is alive. Although this is an optional feature, it's strongly suggested that the device should implement it.

2.1.5 R_RESET_DEV – Reset device and save non-volatile registers

Bit	7	6	5	4	3	2	1	0
Add: 011	**BOOT_EE**	**BOOT_DEF**	-	-	-	**SAVE**	**RST_EE**	**RST_DEF**

- **Bits 0 – RST_DEF**

If this bit is written t to 1, the device resets and boots with all the registers, both Common and Application registers, with the default values. EEPROM will be erased, and the default values will be the permanent boot option.

This bit is always read as 0.

- **Bits 1 – RST_EE**

If this bit is written to 1, the device resets and boots with all his registers, both Common and Application registers, with the values saved on the non-volatile memory, usually an EEPROM. The EEPROM values will be the permanent boot option.

Should not be possible to write to this bit if the non-volatile memory is empty.

This bit is always read as 0.

- **Bits 2 – SAVE**

If this bit is written to 1, the device saves all the non-volatile registers (Common and Application) to the internal non-volatile memory and boots. The non-volatile memory should be the permanent boot option.

This bit is always read as 0.

- **Bits 3 – NAME_TO_DEFAULT**

If this bit is written to 1, the device boots with the default name.

This bit is always read as 0.

- **Bits 6 – BOOT_DEF**

It is a state bit (read only). Indicates that the device booted with the default register values.

- **Bits 7 – BOOT_EE**

It is a state bit (read only). Indicates that the device booted with the register values saved on the EEPROM.

Note: To avoid unexpected behaviors, one bit at a time should be written to register R_RESET_DEV.

2.1.6 R_DEVICE_NAME

An array of 25 bytes that should contain the device name.

The last byte and the bytes not used must be equal to 0.

This register is non-volatile. The device will reset if this register is written.

2.1.7 R_SERIAL_NUMBER – Device's serial number

Add: 000	**SERIAL_NUMBER [15:0]**

This number should be unique for each unit of the same Device ID.

To write to this register a two-step write command is needed. First, write the value 0xFFFF, and then the desired serial number. The device will reset after the second write command is sent.

2.1.8 R_CLOCK_CONFIG – Synchronization clock configuration

Bit	7	6	5	4	3	2	1	0
Add: 014	**CLK_LOCK**	**CLK_UNLOCK**	-	**GEN_ABLE**	**REP_ABLE**	-	**CLK_GEN**	**CLK_REP**

- **Bits 0 – CLK_REP**

If this bit is written t to 1, the device will repeat the Harp Synchronization Clock to the Clock Output connector, if available. It will act has a daisy-chain by repeating the Clock Input into Clock Output.

Writing 1 to this bit also unlocks the Harp Synchronization Clock.

This bit is read as 1 if the device is repeating the Harp Synchronization Clock.

- **Bits 1 – CLK_GEN**

If this bit is written to 1, the device resets will generate the Harp Synchronization Clock in the Clock Output connector, if available. The Clock Input is ignored.

This bit is read as 1 if the device is generating the Harp Synchronization Clock.

- **Bits 3 – REP_ABLE**

This is a read only bit. Written to this bit will not have any effect.

The bit is equal to 1 if the device can repeat the Harp Timestamp Clock.

- **Bits 4 – GEN_ABLE**

This is a read only bit. Written to this bit will not have any effect.

The bit is equal to 1 if the device can generate the Harp Timestamp Clock.

- **Bits 6 – CLK_UNCLOCK**

If this bit is written to 1, the device will unlock the timestamp register counter (register R_TIMESTAMP_SECOND) and will accept new timestamp values.

This bit is read as 1 if the timestamp register is unlocked.

- **Bits 7 – CLK_LOCK**

If this bit is written to 1, the device will lock the current timestamp counter (register R_TIMESTAMP_SECOND) and will not be able to accept new timestamp values.

This bit is read as 1 if the timestamp register is locked.

Note: The device always wakes up in the unlock state.

Version Control

V0.2
First draft released.

V1.0
R_RESET_DEV and **R_DEVICE_NAME** are now optional.
Changed **Normal Mode** to **Standby Mode**.
Added bit **ALIVE_EN** to register **R_OPERATION_CTRL**. This is an important feature.
Major release.

V1.1
Added bit **MUTE_RPL** to register **R_OPERATION_CTRL**.

V1.2
Corrected some wrong names.

V1.3
Added the bit **NAME_TO_DEFAULT**.

V1.4
Added the register **R_SERIAL_NUMBER**.

V1.5
Added the register **R_CLOCK_CONFIG**.

V1.6
Changed device naming to Controller and Peripheral.