Summary - Matchstiq_Z1_GP_Out

Name	matchstiq_z1_gp_out
Version	v1.5
Release Date	5/2019
Component Library	ocpi.assets.platforms.matchstiq_z1.devices
Workers	matchstiq_z1_gp_out.hdl
Tested Platforms	xsim, matchstiq_z1

Revision History

Revision	Description of Change	Date
v1.5	Initial Release	5/2019

Functionality

The Matchstiq-Z1 GP Out device worker controls the three GPIO pins, FPGA_GPIO1, FPGA_GPIO2, and FPGA_GPIO3 present on the Matchstiq-Z1 platform. It is configured for general purpose output.

Worker Implementation Details

The Matchstiq-Z1 GP Out device worker can control the GPIO pins via the property `mask_data`, it's (optional) input port, and the devsignal, `dev_gp`. The devsignal only controls FPGA_GPIO1, while the property and the (optional) port control all 3 pins.

The (optional) input port uses a protocol that has one opcode and contains data and a mask. See `gpio_protocol.xml` in `ocpi.core/specs/gpio_protocol.xml` for further details on the protocol.

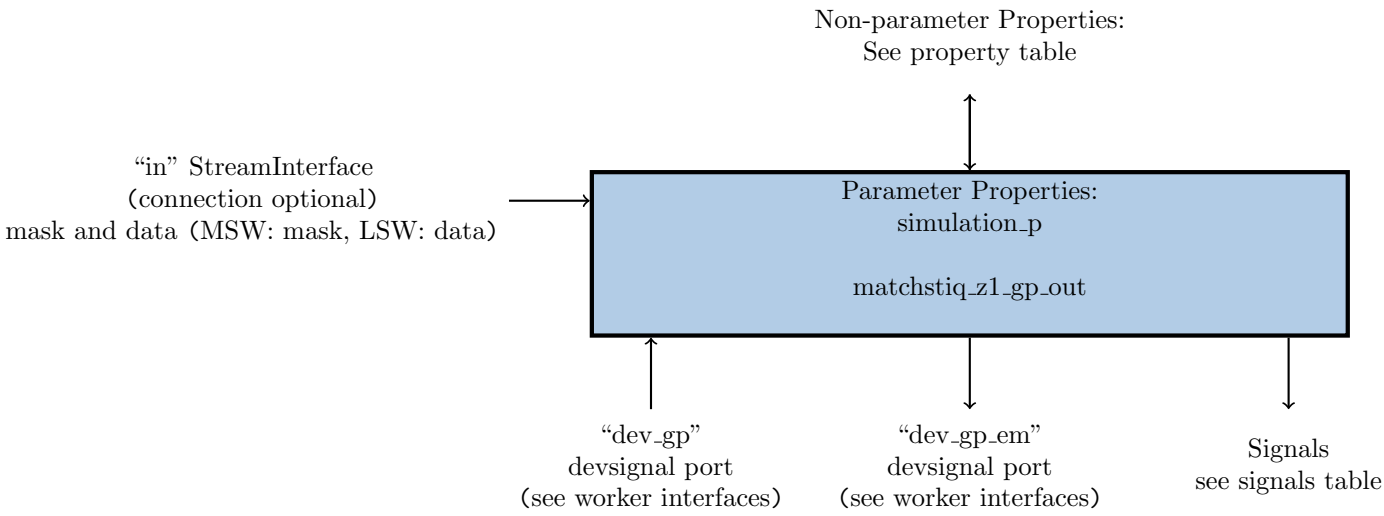
The devsignal `dev_gp` is controlled by the lime_tx device worker through the lime_tx's `dev_txen_dac_in.txen` devsignal. This provides a way to control a device when transmit is enabled. By default FPGA_GPIO1 is controlled by lime_tx. When the Matchstiq-Z1 GP Out device worker is in the reset state, this signal is still controlled by lime_tx.

All three ways of controlling the GPIO pins set the values of pins using a data and mask. In order for a GPIO pin to take on the value of a set data bit, the corresponding mask bit has to be set. The MSW of `mask_data` and the data port data, must be the mask and LSW must be the data and the 3 LSB of the mask and data correspond to the three GPIO pins of the Matchstiq-Z1.

The `input_mask` property provides a way for enabling or disabling the use of the property `"mask_data"`; in port "data" and "mask"; or the devsignal "data" and "mask", if it desired to enable or disable any one of these ways of controlling the GPIO pins. By default all three are enabled.

Block Diagrams

Top level



Source Dependencies

matchstiq_z1_gp_out.hdl

- assets/hdl/platforms/matchstiq_z1/devices/matchstiq_z1_gp_out.hdl/matchstiq_z1_gp_out.vhd

matchstiq_z1_gp_out_em.hdl

- assets/hdl/platforms/matchstiq_z1/devices/matchstiq_z1_gp_out_em.hdl/matchstiq_z1_gp_out_em.vhd
- assets/hdl/primitives/misc_prims/misc_prims_pkg.vhd
  assets/hdl/primitives/misc_prims/edge_detector/src/edge_detector.vhd

Worker Properties

Name	Type	Default	SequenceLength	ArrayLength	ArrayDimensions	Parameter	Accessibility	Usage
input_mask	uChar	0x07	-	-	-	false	Writable	Bitfield that allows enabling or disabling the use of the property "mask_data"; in port "data" and "mask"; or the devsignal "data" and "mask". Bit 0 is the property, bit 1 is the in port, and bit 2 is the devsignal. If a bit is a 1 then the corresponding way of controlling the GPIO pin is enabled.
mask_data	uLong	0	-	-	-	false	Volatile, Writable	Bitfield containing the data to write the GPIO pins and the mask. The mask allows setting GPIO pins on or off in a single operation. The MSW must be the mask and LSW must be the data and the 3 LSB of the mask and data correspond to the 3 GPIO pins of the Matchstiq-Z1. For example if mask_data = 0x00010003, the mask = 0x0001 and data = 0x0003. This would set gpio1 to 1.
simulation_p	bool	false	-	-	-	true	Readable	If true generate circuits for simulation logic.

Component Ports

Name	Protocol	Producer	Optional	Usage
in	gpio_protocol	false	true	Data to write GPIO pins

Worker Interfaces

matchstiq_z1_gp_out.hdl

Type	Name	DataWidth (b)	Advanced	Usage
StreamInterface	in	32	-	Data to write GPIO pins

Type	Name	Count	Optional	Master	Signal	Direction	Width	Description
DevSignal	dev_gp	1	True	False	data	Output	1	Controlled by dev_txen_dac_in.txen within the lime_tx device worker. Used to control GPIO1 pin.
					mask	Output	1	Controlled by dev_txen_dac_in.txen within the lime_tx device worker. Used in logic for controlling FPGA_GPIO1 pin.
DevSignal	dev_gp_em	1	False	True	enable	Output	1	Controls when the matchstiq_z1_gp_em device emulator should start sending messages.

Signals

Name	Type	Width (b)	Description
gpio1	out	1	Output to GPIO pins
gpio2	out	1	Output to GPIO pins
gpio3	out	1	Output to GPIO pins

Control Timing and Signals

The Matchstiq-Z1 GP Out device worker uses the clock from the Control Plane and standard Control Plane signals.

Worker Configuration Parameters

Matchstiq_Z1_GP_Out.hdl

Table 1: Table of Worker Configurations for worker: matchstiq_z1_gp_out

Configuration	simulation_p
0	false
1	true

Performance and Resource Utilization

Matchstiq_Z1_GP_Out.hdl

Table 2: Resource Utilization Table for worker "matchstiq_z1_gp_out"

Configuration	OCPI Target	Tool	Version	Device	Registers (Typ)	LUTs (Typ)	Fmax (MHz) (Typ)	Memory/Special Functions
0	stratix4	Quartus	17.1.0	N/A	235	138	N/A	N/A
0	zynq	Vivado	2017.1	xc7z020clg400-3	235	222	N/A	N/A
0	zynq_ise	ISE	14.7	7z010clg400-3	232	331	446.183	N/A
0	virtex6	ISE	14.7	6vcx75tff484-2	232	331	352.583	N/A
1	stratix4	Quartus	17.1.0	N/A	237	143	N/A	N/A
1	zynq	Vivado	2017.1	xc7z020clg400-3	236	224	N/A	N/A
1	zynq_ise	ISE	14.7	7z010clg400-3	233	332	445.071	N/A
1	virtex6	ISE	14.7	6vcx75tff484-2	233	332	351.74	N/A

Test and Verification

The matchstiq_z1_gp_out device worker unit has three test cases and utilizes the matchstiq_z1_gp_out_em device emulator. Case 1 tests controlling the GPIO pins via the `mask_data` property, case 2 tests controlling them via the input port, and case 3 tests controlling via the `dev_gp` devsignal.

For the mask_data property and the data port, the tests that are done are: setting data bits high but not setting masks, setting data bits high and setting the appropriate mask bits high, clear data by setting data to 0x0000 and mask to 0x0007, and then set data bits high but only set some of the appropriate masks high.

For the devsignal the mask is held high but the data is toggled on and off.

The `generate.py` script generates the input data and generates golden data files. For case 1 and 3 the input file is a 0 byte file since the input port data is ignored in these two cases. A `.golden.data` file is generated for each case.

The `verify.py` script checks that the output data matches the expected output data contained in the `.golden.data` files.