The Total CPU (TCPU) Description

TCPU is easily realizable and easily modified RISC processor mainly aimed for implementation in FPGAs. The key feature of this CPU is that its command set does not dependent upon data word width, and, therefore, minimal modifications are required for code reuse. The CPU is realized on Verilog-2001 and easily expanded to include additional modules and commands. TCPU doesn't require a lot of FPGA resources. For example, minimal 12-bit realization on Cyclone FPGA (Altera) requires 167 LCs, 32-bit realization – 314 LCs.

The TCPU consists of 3 main parts. They are general purpose register blocks, ALU and control unit. There are also special bitwise implemented registers, such as the status register. They are picked out from general purpose blocks because their bits are used (and can be modified) during execution of the operations. PC can be implemented either in general purpose block or as a special register. The simplified data paths of these two variants are shown on fig. 1,2.

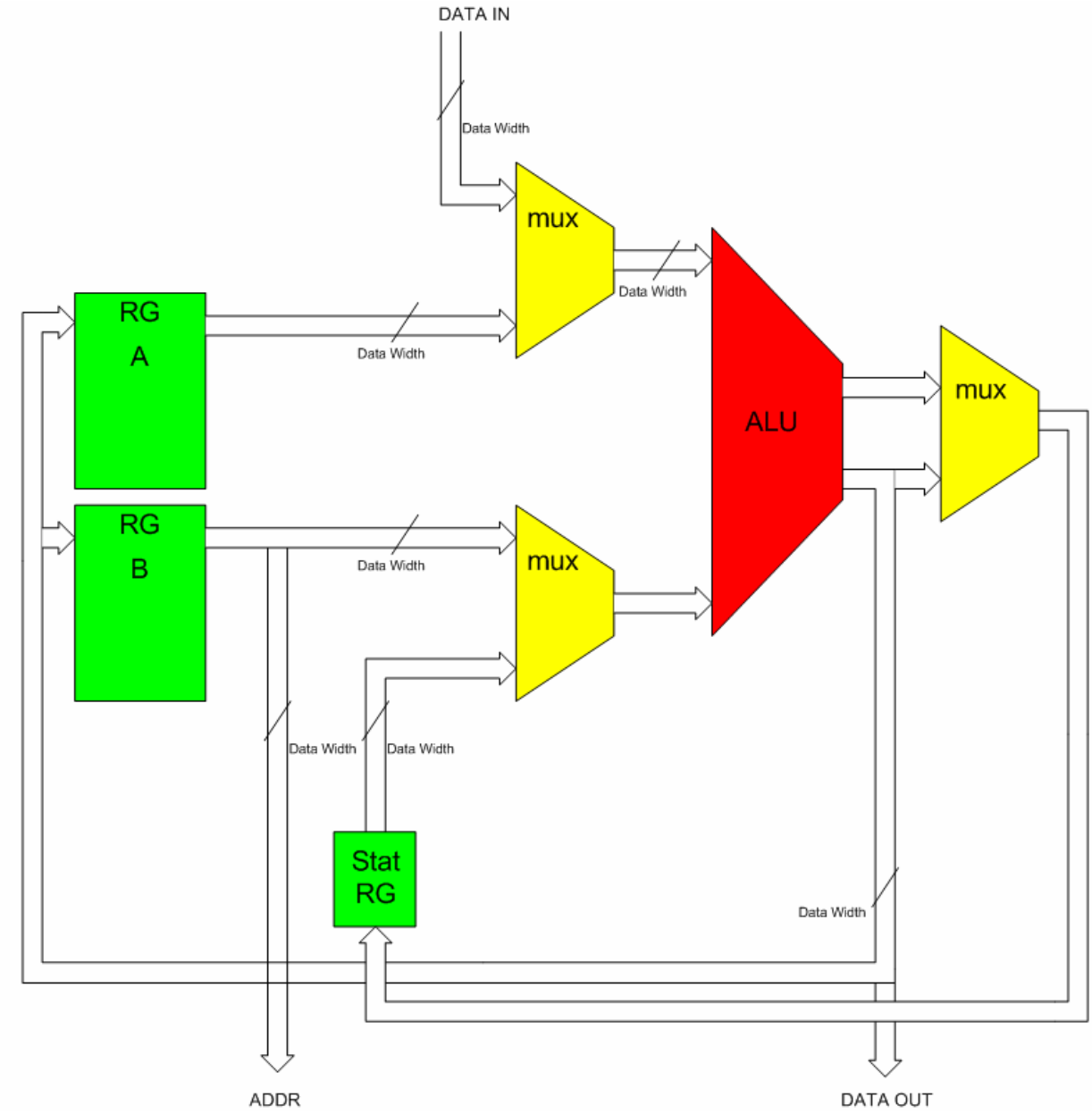

Fig. 1.

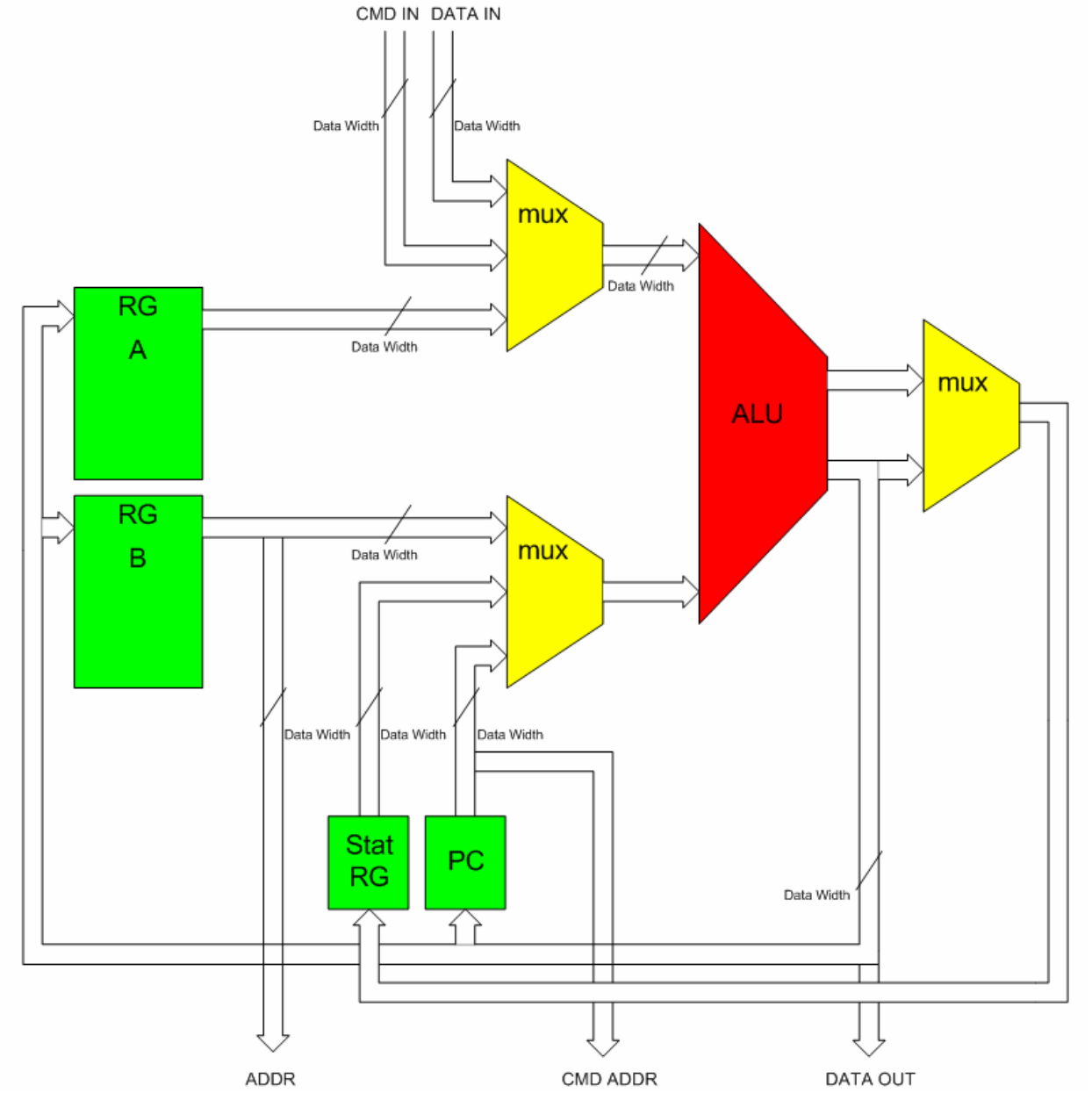

Fig. 2.

The first figure corresponds to PC implemented in general purpose register block, and the second – to PC implemented as a standalone register. The first variant requires less hardware resources but it is almost 2 times slower then the second variant. Register blocks RG A and RG B contain the same values in order to fasten the CPU operation. ALU performs logical and arithmetic operations. All the co-ordination of CPU parts is performed by the control unit. If hardware PC is implemented, there is also additional multiplexer on the PC data input which switches it either to the ALU output or directly to the output of RG-B register block. This allows one-cycle interrupt entering and simplifies returning from interrupt. Hardware requirements for various TCPU implementations are presented in table 1.

Type / Data Width	12 bit	16 bit	24 bit	32 bit
Cyclone, SPC[1]	217 LCs	262 LCs	353 LCs	444 LCs
Cyclone	167 LCs	197 LCs	256 LCs	314 LCs
Virtex, SPC[1,2]	209 Slices	254 Slices	350 Slices	450 Slices
Virtex[1,2]	139 Slices	165 Slices	220 Slices	269 Slices

Table 1.

[1] - means PC implementation as a special register.

[2] - means ALU implementation using behavioral synthesis.

The command set of TCPU is shown in table 2.

Command / bits	Second word	11-8	7-4	3-0
XOR	–	COP (0)	DST	SRC
AND	–	COP (1)	DST	SRC
OR	–	COP (2)	DST	SRC
NOT	–	COP (3)	DST	–
ADD	–	COP (4)	DST	SRC
SUB	–	COP (5)	DST	SRC
SHR	–	COP (6)	DST	–
INR	–	COP (7)	DST	–
MOV	–	COP (8)	DST	SRC
L	–	COP (9)	DST	ADR RG
S	–	COP (A)	SRC	ADR RG
LS	–	COP (B)	DST	SRC-S
SS	–	COP (C)	SRC	DST-S
MVRC	SRC-C	COP (D)	DST	CND
ADRC	SRC-C	COP (E)	DST	CND
Reserved	*	COP (F)	*	*

Table 2.

In table 2 COP denotes the code of operation, DST – destination register, SRC – source register, DST-S – special destination register, SRC-S – special source register, SRC-C – source from code memory, ADR RG – register containing address of the operand, CND – condition code. There are 8 condition codes defined allowing adding 8 custom conditions (table 3).

Condition code	Mnemonic	Meaning
0	ALLW	Always true
1	Z	True if Z is set
2	C	True if C is set
3	O	True if O is set
4	N	True if N is set
5	NZ	True if Z is cleared
6	NC	True if C is cleared
7	CN	True if both C and N are set

Table 3.

The meanings of the mnemonics are:

- XOR – exclusive bitwise OR;
- AND – bitwise AND;
- OR – bitwise OR;
- NOT – bitwise negation;
- ADD – signed addition;
- SUB – signed subtraction;
- SHR – arithmetic rotation for 1 bit right;
- INR – increment;
- MOV – must be clear for everyone who reads this ☺;
- L – load register from external memory;
- S – save register into external memory;
- LS – load register from special register;
- SS – save register into special register;
- MVRC – conditional move of a constant from program memory into a register;
- ADRC – conditional addition of a constant from program memory into a register;

So, there are 9 commands that operate with general purpose registers (XOR,AND,OR,NOT,ADD,SUB,INR,SHR,MOV), 2 commands for data exchanging with external memory (L,S), 2 commands for data exchanging with special registers (LS,SS), and 2 commands for moving or adding constants from program memory into registers. The last 2 commands are also used as absolute or relative conditional jumps if the field DST=0. So, in the case of PC implementation as a standalone register, commands MVRC 0,CND and ADRC 0,CND affect PC but not the register RG0. There is also another exclusion – if bit 3 in the CND field is set to 1, contents of RG1 are written into PC during these commands. It doesn't affect the execution of the command and is used as two-in-one command: conditional "return from interrupt" and "move constant to register" commands. In the case when PC is the general purpose register, the command ADRC should be used only with PC in consequence of hardware restrictions.

According to command set, TCPU has 16 general purpose registers visible at a time (RG0-15), and 16 special registers (SR0-15). SR0 is always reserved as a PC. If PC is implemented in general purpose blocks, this register shouldn't be used. SR1 is a status register. Its contents are shown on table 4.

Bit	5...	4	3	2	1	0
Meaning	Reserved	I	C	O	N	Z

Table 4.

The bit I is the interrupt enable flag. The destination of other flags is clear. The commands modifying the status register bits 3..0 are presented on the table 5.

Command / flag	N	C	O	Z
XOR	●	×	×	●
AND	●	×	×	●
OR	●	×	×	●
NOT	●	×	×	●
ADD	●	●	●	●
SUB	●	●	●	●
INR	●	●	●	●
SHR	●	×	×	●

Table 5.

Here the sign "●" means that the flag is modified during an operation, "**×**" means that the flag is modified, but should not be used.

It is clear from table 2 that the command set contains only two formats – with or without second word. It is considerably simplifies the control unit and, therefore, saves hardware resources.

So, the command word of TCPU contains 12 bits independently from data path width. There are 4 idle bits in commonly used 16-bit variant. They can be used for implementing additional commands or for extending the TCPU functionality. Despite of the fact that command word width is fixed, the external command bus width is equal to data path width, and the second word of two-word commands use all its bits to contain operand SRC-C. If the command bus width is more than 24 bits, an external circuit can be used to extract the commands either from higher or from lower part of the bus. An example of such a circuit is presented here, and it's connection to the TCPU is shown on fig. 3. It requires only 12 Altera LCs for implementation.

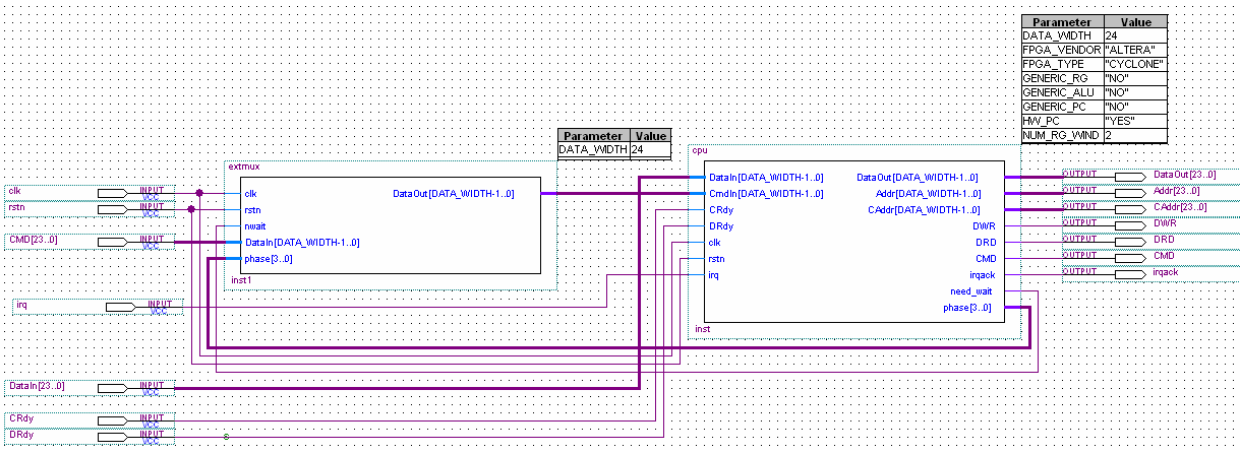

Fig. 3.

The TCPU has a register-based interrupt system, which is this: when the interrupt comes, the contents of the PC and RG1 are interchanged causing jump to the interrupt vector and at the same time saving the PC. The flags should be saved and restored before exit by interrupt handler procedure. So, in TCPU RG1 must contain a valid interrupt vector before enabling interrupts. Returning from interrupts is implemented in TCPU as a MVRC command with bit 3 of CND field set to 1. In further versions the interrupt system will use register windows.

There are some definitions and parameters that influence the compilation of TCPU. They are presented on table 6.

Control structure	Realization	Meaning
GENERIC_ALU	Definition	Whether ALU is implemented using behavioral synthesis
GENERIC_PC	Definition	Whether PC is implemented using behavioral synthesis
HW_PC	Definition	Is the hardware PC used
GENERIC_RG	Definition	Whether register blocks are implemented using behavioral synthesis
DATA_WIDTH	Parameter	Data path width
FPGA_VENDOR	Parameter	Vendor of destine FPGA
FPGA_TYPE	Parameter	FPGA family
NUM_RG_WIND	Parameter	Number of register windows (reserved)

Table6.

The TCPU has the following external signals:

SIGNAL	**WIDTH**	**DIRECTION**	**ACTIVE LEVEL**	**DESCRIPTION**
DataIn	DATA_WIDTH	In		Data Memory input
DataOut	DATA_WIDTH	Out		Data Memory input
Addr	DATA_WIDTH	Out		Data Memory address
CmdIn	DATA_WIDTH	In		Command Memory input*

CAddr	DATA_WIDTH	Out		Command Memory address*
CRdy	1	In	High	Command Memory ready*
DRdy	1	In	High	Data Memory ready*
DWR	1	Out	High	Data Memory write enable
DRD	1	Out	High	Data Memory read enable
CMD	1	Out	High	Command Memory write enable
irq	1	In	High	Interrupt request input
irqack	1	Out	High	Interrupt acknowledge
nextphase	4	Out		Next phase of the core
phase	4	Out		Present phase of the core
clk	1	In	Rising edge	Input clock
rstn	1	In	Low	Negative reset

* - signals are generated when “SPLIT_BUSES” is defined.

Table 7.

Actually, there are two variants of realization of ALU, PC and registers for now. The first is intended for implementation on Altera FPGAs. It uses vendor-defined megafunctions and is optimized for Altera’s logical cells. ALU in this variant processes each operation in 4 stages. At the first stage, depending on operation code, operand A (fig. 1-2) is inverted, cleared or passed without modification. At the second stage the logical operations NOT, AND, OR, XOR are performed. At the third stage addition and subtraction are performed, and at the last stage the result can be shifted right for 1 bit. The second variant uses behavioral synthesis and can be implemented in large number of FPGAs, though it requires slightly more resources.