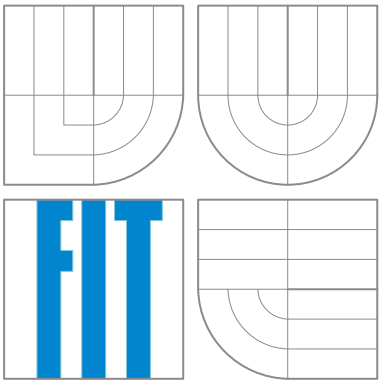


Hardware/Software Codesign

Vestavěný systém pro filtraci a segmentaci obrazu

2. ledna 2014

Autoři: Pavel Macenauer, xmacen02

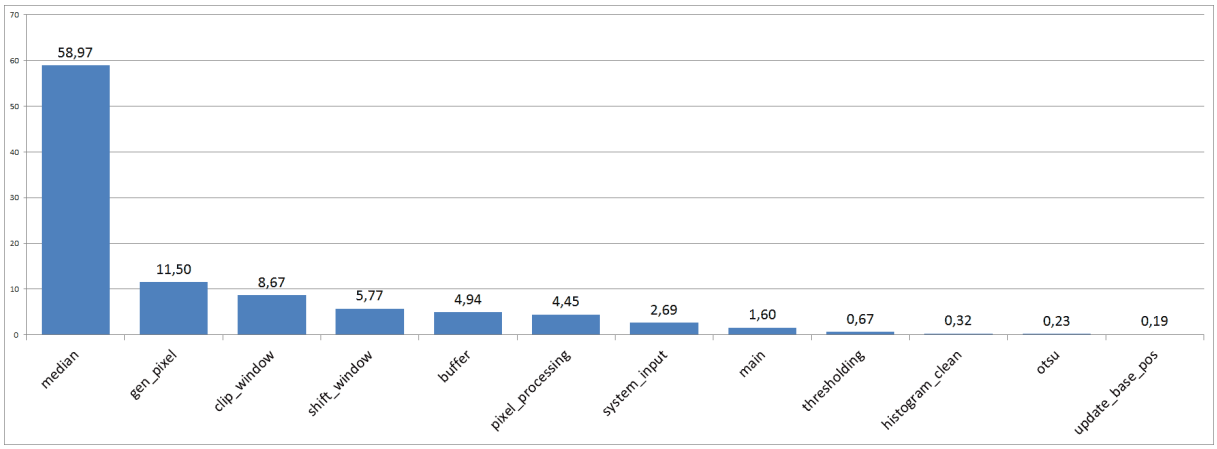
Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

Obsah

Analýza algoritmu z programu gprof

Čas [%]	Jméno funkce
58,97	`median`
11,5	`gen_pixel`
8,67	`clip_window`
5,77	`shift_window`
4,94	`buffer`
4,45	`pixel_processing`
2,69	`system_input`
1,6	`main`
0,67	`thresholding`
0,32	`histogram_clean`
0,23	`otsu`
0,19	`update_base_pos`

Tabulka 1: Podíl celkového času běhu programu pro jednotlivé funkce



Obrázek 1: Podíl celkového času běhu programu pro jednotlivé funkce

Rozdělení aplikace mezi software a hardware

Původní funkce na MCU	Přesunuto na HW
otsu	**NE**
median	ANO
buffer	ANO
clip_window	ANO
shift_window	ANO
system_input	ANO
histogram_clean	**NE**
thresholding	ANO
print_results	**NE**
pixel_processing	ANO
update_base_pos	ANO, realizováno komponentou Generátor pixelů
gen_pixel	ANO, realizováno komponentou Generátor pixelů

Tabulka 2: Rozdělení aplikace mezi software a hardware

Využití adresového prostoru sdílené paměti

Adresa	Jméno	Poznámka	MCU
0	`MCU_DATA_HIST0`	První položka histogramu	vstup
1	`MCU_DATA_HIST1`		vstup
2	`MCU_DATA_HIST2`		vstup
3	`MCU_DATA_HIST3`		vstup
4	`MCU_DATA_HIST4`		vstup
5	`MCU_DATA_HIST5`		vstup
6	`MCU_DATA_HIST6`		vstup
7	`MCU_DATA_HIST7`	Poslední položka histogramu	vstup
8	`MCU_DATA_THRESHOLD`	MCU zde ukládá nově vypočtený práh, FPGA si ho načítá	výstup
9	`MCU_FLAG_OTSU_START`	Příznak, který určuje, že se má vypočíst nová hodnota prahu	vstup

Tabulka 3: Využití adresového prostoru sdílené paměti

Vlastnosti obvodu uvnitř FPGA

- Inicializační interval smyčky `(PIPELINE_INIT_INTERVAL)` 4
- Latence obvodu 4 (při periodě 40 ns)

Number of Slice Flip Flops	394 out of 1,536	25%
Number of 4 input LUTs	914 out of 1,536	59%
Number of occupied Slices	610 out of 768	79%
Total Number of 4 input LUTs	1,002 out of 1,536	65%
Number used as logic	914	
Number used as a route-thru	88	

Tabulka 4: Využití a rozdělení obvodu

Porovnání softwarové a rozdělené implementace

	Software	Rozdělená aplikace
Doba zpracování pixelu [μs]	182	0,16
Zpracovaných pixelů za sekundu [px/s]	5494	6250000
Zrychlení	1x	1137x

Tabulka 5: Porovnání čistě softwarové a rozdělené aplikace mezi hardware a software

Shrnutí

Ukázalo se, že čistě softwarová implementace je v praxi nepoužitelná, protože řešení vygeneruje 0,07 framů za sekundu (fps) při rozlišení 320x240. Při použití FPGA (hardwaru) se aplikace výrazně zrychlila a byla schopná vygenerovat 81 fps, což splňuje požadavek minimálně 60 fps.

Pravděpodobně by šlo přesunout i výpočet prahu metodou otsu z MCU na FPGA, ale výsledek by to mělo pouze na výslednou kvalitu obrazu, protože bychom byli schopni aktualizovat práh častěji. Není to tak potřeba a může probíhat víceméně asynchronně na MCU.

Úzkým hrdlem mi přišlo především omezení počtu zápisu a čtení do/ze sdílené paměti, kdy se musely redukovat veškeré pomocné příznaky na minimum. Pro složitější algoritmy zpracování obrazu, využívající více zdrojů, by se tak musel výrazně zvýšit inicializační interval smyčky.