

پروژه نهایی درس طراحی سیستم های کامپیوتری دیجیتال

محمد حسين ميرزائي -99522158

متين محمودخاني-99522095

نام استاد : دكتر بيت اللهي

این پروژه به طور کلی به پردازش تصویر می پردازد و از چند قسمت کلی تشکیل شده که در ادامه توضیح داده خواهد شد .

ابتدا تصویری را در نظر گرفته و آن را سیاه سفید می کنیم (مود grey scale) ، چون که تصویر های رنگی سه کاناله بوده و با تصویر سیاه سفید بررسی ما راحت تر خواهد شد .

سپس با استفاده از کتاب خانه های مورد نظر در زبانی برنامه نویسی نرم افزار نظیر پایتون ، عکس مورد بررسی را پیکسل بندی کرده و تبدیل به آرایه می کنیم ، حال آرایه بدست آمده را به عنوان داده به زبان برنامه نویسی سخت افزار برای پردازش می دهیم .

بخش سخت افزار نیز به سه بخش main , package , simulate تقسیم بندی شده است که به توضیح هر یک می پردازیم :

<u>Package</u>: ابتدا کتابخانه IEEE را اضافه می کنیم ،در این بخش اطلاعات کلی را ذخیره سازی و از آن ها استفاده خواهیم کرد ، به ترتیب بخش فیلتر ها که هر فیلتر از یک آرایه دو بعدی real تشکیل شده است (مثلا 3%3)،تایپی برای گرفتن عکس اصلی ما که با توجه به گفته سوال دو بعدی و از نوع integer و (مثلا 128*32 می باشد ،و قسمت بعدی نیز عکس پردازش شده توسط فیلتر ها است که به صورت آرایه دو بعدی از نوع real می باشد و سایز آن به دلیل سایز عکس و سایز فیلتر 126*126 می باشد .

تصوير نظير،

نحوه طی شده 6 فیلتر و دلیل Kernel size: 3 آن که عکس Padding:-0 پردازش شده Dilation: 126 * 126 1 Stride:-می باشد را نشان می دهد: Output (4×4) :

حال با توجه به تایپ های تعریف شده عضوی constant برای ورودی عکس که آرایه دو بعدی در نظر گرفته شد.

و در نهایت در بخش function، package برای گرفتن عده random تعریف می شود ، در بخش package body نیز این function پیاده می شود که بدین صورت می باشد که با اضافه کردن کتاب خانه package body بخش MATH_REAL با فراخوانی uniform ، procedure عددی انتخاب و با توجه به بازه مورد نظر عدد random انتخاب می شود(دقت شود که فیلتر ما شامل اعداد رندوم بین صفر و یک می باشد ، پس بازه مورد بررسی را بین صفر و یک در نظر می گیریم .)

eneric ، n و بخش به پیاده سازی کلی کد می پردازیم ، 1 ورودی clock و خروجی ما عکس پردازش شده می باشد و یک عدد generic ، n برای تعداد فیلتر ها و پیاده سازی کلی در نظر می گیریم .

حال در این بخش ورودی از تایپ all_filters که برای گرفتن تمام فیلتر ها می باشد ، یک counter از نوع integer و یک result و یک result و یک

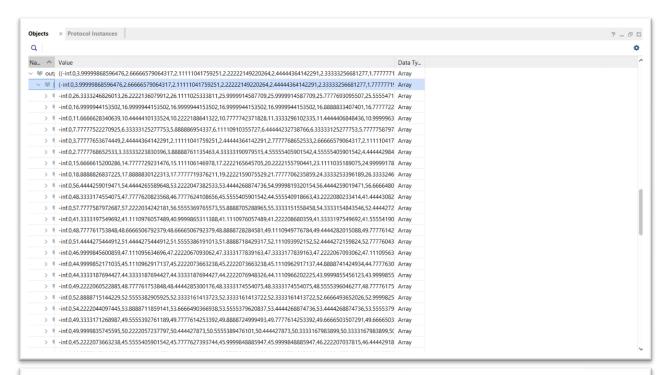
در بخش اول ، ابتدا به صورت concurrent و با استفاده از کلید واژه generate یک حلقه تو در تو ایجاد و تمام فیلتر ها را می گیریم ، در قسمت بعدی نیز با توجه به توضیحات مسئله، دو process پیاده می کنیم .

Process ابتدایی به پیاده سازی ردیف به ردیف عکس پردازش شده می پردازد و به مقدار counter حساس می باشد و با تغییر convolution ، ردیف های متفاوت را اجرا می کند و convolution این بخش نیز طبق گفته سوال به صورت درختی و با در نظر گرفتن هشت مقدار temp در هر بخش انجام شده است .

Process دوم نیز با توجه به clock عمل می کند و هر بار که ما rising_clk داریم مقدار counter را اضافه می کند ، دلیل استفاده از clock آن است که سطر به سطرطبق لبه بالا رونده clock پیش برویم و حجم عظیم داده به یک دفعه سرازیر نشود .

Simulate : در این بخش با مقدار دهی کلاک به صورت منظم و port map بخش main می توان داده های نهایی را شبیه سازی کرد(دقت شود به علت استفاده از تایپ real امکان سنتز کد وجود نداشته و صرفا شبیه ساز آن را در اختیار داریم .)

نمونه ای از خروجی شبیه سازی شده در صفحه بعد قابل رویت می باشد:



Data Tv... Logic .3.99999868596476.2.66666579064317.2.11111041759251.2.22222149220264.2.44444364142291.2.33333256681277.1.7777771 Array 3.99999868596476,2.66666579064317,2.11111041759251,2.2222149220264,2.44444364142291,2.33333256681277,1.7777771! Array .99999868596476.2.66666579064317.2.11111041759251.2.2222149220264.2.44444364142291.2.33333256681277.1.77777719 Array .:66666579064317,3.4444433129141,3.99999868596476,4.4444298440529,4.4444298440529,4.55555405901542,4.44442984 Array 1.9999960578943.12.7777735801652.14.2222175500969.16.6666611915198.19.4444380567731.22.5555481458568.24.8888807 Array i3.3333158128634,51.9999829175418,49.7777614253392,47.7777620823568,47.5555399331366,49.1110949776784,51.8888718 Array l8.3333174554075.47.111095634696.44.3333187694427.40.7777643819185.37.9999875166652.36.7777656959537.37.11109891 Array 7.8888698718788,56.4444259019471,53.8888711859141,51.1110943206608,49.2222060522885,48.6666506792379,48.8888728 Array 0.7777643819185,40.6666533073084,40.6666533073084,40.6666533073084,40.6666533073084,40.7777643819185,41.1110976 Array 8.6666506792379.48.6666506792379.48.6666506792379.48.6666506792379.48.777761753848.48.8888728284581.49.33331712 Array 1.6666496937114,51.6666496937114,51.6666496937114,51.6666496937114,51.7777607683216,51.8888718429317,52.3333161 Array l6.9999845600859.46.9999845600859.46.9999845600859.46.8888734854758.46.7777624108656.46.6666513362555.46.6666513 Array 4.6666519932731.44.6666519932731.44.6666519932731.44.555540918663.44.4444298440529.44.3333187694427.44.33331876 3.9999855456123.43.9999855456123,43.9999855456123,43.8888744710022,43.7777633963921,43.7777633963921,43.8888744 Array 9.5555392761189,49.1110949776784,48.777761753848,48.6666506792379,48.6666506792379,48.8888728284581,49.11109497 i2.8888715144229,52.5555382905925,52.3333161413723,52.3333161413723,52.6666493652026,52.9999825 Array 4.2222044097445,53.8888711859141,53.6666490366938,53.55553796208: 7,53.4444268: 374736,53.4444268874736,53.5555379 Array اج.1110949776784,49.2222060522885,49.4444282015088,49.55553927611(كرية 1,49.444424) كالمرابق المرابق i0.1110946491696,50.2222057237797,50.444427873,50.5555389476101,50.444427873,50.3333167983899,50.3333167983899,50 15.3333184409339,45.5555405901542,45.7777627393744,45.9999848885947,45.9999848885947,46.222207037815,46.44442918 Array 9.3333204119868.39.3333204119868.39.4444314865969.39.555542561207.39.555542561207.39.555542561207.39.4444314865