راهنما: استیت ها با رنگ قرمز و سیگنال‌ها با رنگ سبز مشخص شده اند.

ماژول Interpreter

وظیفه‌ی این ماژول تفسیر داده‌ی دریافتی از رابط سریال است. این ماژول در ابتدای بازی عرض زمین را گرفته و در خروجی خود قرار می‌دهد(سیگنال width) و سیگنال width_valid را به نشانه‌ی معتبر بودن مقدار عرض زمین در کل مدت بازی یک 1 نگه می‌دارد. برای طول زمین و رنگ بازیکن نیز به همین ترتیب عمل می‌کند. پس از آن ماژول به حالت(state) get_oc رفته و منتظر دریافت کروشه‌ی باز می‌ماند. با دریافت کروشه باز سیگنال opponent_game_start یک سیکل کلاک یک می‌شود. در get_enemy_move در صورتی که کروشه‌ی بسته دریافت شود به get_cc رفته و me_game_start یک سیکل کلاک یک 1 می‌شود. در صورتی که در get_enemy_move کروشه دریافت نشود، بنابراین داور حرکت حریف را ارسال کرده است در نتیجه حرکت حریف دریافت شده و سیگنال new_en_mov یک سیکل کلاک یک 1 می‌شود در حقیقت سیگنال new_en_mov معتبر بودن RxD_data_Opponent را به ماژول مشتری interpreter اعلام می‌کند(در حقیقت ماژول مشتری این ماژول، ماژول Handout است و سیگنال new_en_move به ورودی Rxd_data_opponent_valid ماژول Handout متصل است). بعد از دریافت کروشه بسته ماژول دوباره منتظر دریافت کروشه‌ی باز می‌ماند.

نکته : ورودی RxD_ready که از UART می‌آید تنها یک سیکل یک است.

ماژول memory_handler

وظیفه‌ی این ماژول دریافت جهت و به روز کردن موقعیت فعلی توپ در صفحه و به روز کردن زمین است. پس از دریافت جهت لازم است که نقطه‌ی فعلی و نقطه‌ی بعدی در ram به روز شوند و موقعیت نیز به نقطه‌ی بعدی تغییر پیدا کند(رجیستر second_point برای نشان دادن کار بر روی نقطه‌ی اول و دوم است). ورودی direction_in و direction_valid به خروجی‌های direction_out و direction_valid_out ماژول Handout متصل هستند. با دریافت جهت ورودی به روزرسانی‌های گفته شده انجام می‌شود و سیگنال update_done یک سیکل کلاک یک می‌شود.(خروجی update_done این ماژول به ورودی update_done ماژول Handout متصل است.) و ماشین حالت در get_direction_state منتظر دریافت جهت بعدی باقی میماند. خروجی‌های

current_x و current_y مختصات توپ در نقطه‌ی فعلی را نشان می‌دهند که به ماژول MP متصل هستند. خروجی‌های address و data_out و ورودی data_in نیز به RAM متصل هستند.

روند کار این ماژول به این صورت است که پس از دریافت جهت، اطلاعات خانه‌ی فعلی از RAM خوانده می‌شود و متناسب با جهت تغییر داده می‌شود و در رم ریخته می‌شود. سپس موقعیت توپ به روز شده و همین کار صورت می‌گیرد. از آنجا که آدرس تابعی از current_x و current_y است با به روز رسانی موقعیت توپ آدرس نیز برای دستیابی به خانه‌ی حافظه‌ی نقطه‌ی دوم به روز رسانی می‌شود.

نکته:در ابتدای ماشین حالت طول و عرض زمین دریافت شده و رجیستر می‌شود(سیگنال‌های مربوطه واضح هستند.)

ماژول MP

در این گزارش کمتر وارد جزئیات ماژول MP خواهیم شد. جزئیات کار این ماژول در یک گزارش دیگر به صورت مفصلتر شرح داده خواهد شد. در این ماژول در صورتی که ورودی my_turn یک سیکل ساعت یک شود ماژول شروع به کار کرده و در نهایت جهت را در خروجی direction قرار داده و در صورتی که حرکت بعدی در این جهت نیز با ما باشد سیگنال my_move را یک می‌کند. خروجی‌های direction و my_move زمانی معتبر هستند که خروجی direction_valid یک باشد. خروجی direction_valid برای یک سیکل ساعت یک می‌شود. سیگنال idle بیکار بودن ماژول را نشان می‌دهد. **استراتژی بازی باید در این ماژول پیاده سازی شود.**

ماژول Handout

این ماژول وظیفه دارد تمام ماژول‌های موجود را کنترل کند. در کد بلوک always اول مربوط به ماشین حالت و بلوک های دیگر مربوط به کنترل سایر ماژول هاست که در ابتدای هر قسمت مشخص شده است. در ماشین حالت ابتدا حرکت‌های حریف از interpreter دریافت شده و به memory_handler داده می‌شود تا RAM را به روز کند پس از به روز کردن RAM ماژول MP راه انداخته می‌شود و با دریافت هر حرکت از MP مانند قسمت قبل RAM به روزرسانی می‌شود.این کار تا زمانی که حرکت با ما باشد انجام می‌شود. هر حرکتی که از MP دریافت

می‌شود علاوه بر تحویل به memory_handler در fifo ی مربوط به ارسال نیز ریخته می‌شود. در آخر نیز کروشه بسته و اینتر ارسال می‌شود.