بسمه تعالی

پروژه مدار منطقی بهار ۱۴۰۴

استاد آزمایشگاه : استاد ثاقب حقیقی

محمدرضا حسن‌زاده ۴۰۲۲۶۰۴۶ آرین ابراهیمی ۴۰۲۳۱۰۰۱

مداری که برای این پروژه باید انجام دهیم شامل دو بخش که بخش اول محاسبات مربوط به فرمول ($T = \frac{Cal \times 60}{MET \times W} \times G$) می‌باشد و بخش دوم مربوط به طراحی مدار FSM است. لذا برای انجام فاز اول داریم:

۱. واحد محاسبه (مدار ترکیبی) (Time Calculation Logic) : محاسبه زمان مورد نیاز برای سوزاندن کالری مطلوب

وظیفه اصلی این واحد، محاسبه تعداد کل تمرین‌های یک دقیقه‌ای (T) است که کاربر برای رسیدن به هدف کالری‌سوزی خود نیاز دارد. این محاسبه بر اساس فرمول ارائه شده در صورت پروژه انجام می‌شود:

$$T = \frac{Cal \times 60}{MET \times W} \times G = \frac{Cal \times 60}{W} \times G \times \frac{1}{MET}$$

که در آن:

- Cal: کالری هدف کاربر (ورودی ۲ بیتی برای مقادیر ۵۰، ۱۰۰، ۱۵۰، ۲۰۰ کالری)
- W: وزن کاربر بر حسب کیلوگرم (ورودی ۳ بیتی برای مقادیر ۵۰ تا ۱۲۰ کیلوگرم)
- MET: یکای متابولیکی معادل برای شدت فعالیت (ورودی ۲ بیتی برای مقادیر ۱، ۲، ۴، ۸)
- G: ضریب جنسیت (۱ برای مردان و ۱.۱۲۵ برای زنان طبق تعریف پروژه).

با توجه به محدودیت‌های طراحی (عدم استفاده از عملگرهای ضرب و تقسیم مستقیم در Verilog)، فرمول به صورت مرحله‌ای و با استفاده از جداول جستجو (LUT)، شیفت‌دهنده‌ها، جمع‌کننده‌ها و مالتی‌پلکسرها پیاده‌سازی می‌شود:

جدول درستی عبارت فوق را به صورت زیر می نویسیم :

W = W_2 W_1 W_0, Cal = C_1 C_0, T = time that we calculate for our exercise

W_2	W_1	W_0	C_1	C_0	W (وزن)	Cal (کالری هدف)	T نتیجه
0	0	0	0	0	50	50	60
0	0	0	0	1	50	100	120
0	0	0	1	0	50	150	180
0	0	0	1	1	50	200	240
0	0	1	0	0	60	50	50
0	0	1	0	1	60	100	100
0	0	1	1	0	60	150	150
0	0	1	1	1	60	200	200
0	1	0	0	0	70	50	43 (42.86)
0	1	0	0	1	70	100	86 (85.71)
0	1	0	1	0	70	150	129 (128.57)
0	1	0	1	1	70	200	171 (171.43)
0	1	1	0	0	80	50	38 (37.5)
0	1	1	0	1	80	100	75
0	1	1	1	0	80	150	113 (112.5)
0	1	1	1	1	80	200	150
1	0	0	0	0	90	50	33 (33.33)
1	0	0	0	1	90	100	67 (66.67)

1	0	0	1	0	90	150	100
1	0	0	1	1	90	200	133 (133.33)
1	0	1	0	0	100	50	30
1	0	1	0	1	100	100	60
1	0	1	1	0	100	150	90
1	0	1	1	1	100	200	120
1	1	0	0	0	110	50	27 (27.27)
1	1	0	0	1	110	100	55 (54.55)
1	1	0	1	0	110	150	82 (81.82)
1	1	0	1	1	110	200	109 (109.09)
1	1	1	0	0	120	50	25
1	1	1	0	1	120	100	50
1	1	1	1	0	120	150	75
1	1	1	1	1	120	200	100

- برای پیاده‌سازی این جدول درستی، می‌توان از یک ساختار مبتنی بر گیت‌های منطقی استفاده کرد که با ساده‌سازی هر بیت خروجی (با استفاده از جدول کارنو یا روش‌های دیگر) به دست می‌آید یا همانطور که در کد Verilog پیاده‌سازی شده، از یک بلوک حافظه مقداردهی شده اولیه استفاده کرد.

زیرماژول lut_cal_div_w_unit (مرحله اول: محاسبه $\frac{Cal\times 60}{W}$)

این زیرماژول به صورت کاملاً ترکیبی طراحی شده و وظیفه محاسبه بخش اول فرمول، یعنی $\text{Value1} = \frac{Cal\times 60}{W}$ ، را دارد.

- **نحوه پیاده‌سازی:** برای این محاسبه، از یک جدول جستجو (Look-Up Table - LUT) با نام lookup_table_mem استفاده شده است. این LUT یک آرایه ۳۲ خانه‌ای است که هر خانه آن یک مقدار ۱۰ بیتی را ذخیره می‌کند. انتخاب LUT به دلیل سرعت بالا در محاسبات مبتنی بر جدول و امکان پیاده‌سازی مستقیم جدول درستی ارائه شده در صورت پروژه بوده است.

- **آدرس‌دهی LUT**: ورودی‌های Cal_param (۲ بیت) و W_param (۳ بیت) با یکدیگر الحاق شده ({Cal_param, W_param}) و یک آدرس ۵ بیتی (table_address) را تشکیل می‌دهند. این آدرس برای انتخاب یکی از ۳۲ مقدار ذخیره شده در LUT استفاده می‌شود.
- **محتوای LUT**: هر خانه از LUT حاوی مقدار از پیش محاسبه شده و گرد شده عبارت (مقدار واقعی کالری × ۶۰) / مقدار واقعی وزن برای ترکیب متناظر ورودی‌ها است. این مقادیر در یک بلوک initial در کد Verilog مقداردهی اولیه شده‌اند.
- **خروجی**: خروجی ۱۰ بیتی result_data مقدار خوانده شده از LUT است.

- برای محاسبه بخش دوم ($G \times$)، می توان از یک مالتی پلکسر 2×1 برای انتخاب ضریب (۱ مردان) یا ضریب (۱.۱۲۵ برای زنان) استفاده کرد. برای محاسبه ضریب (۱.۱۲۵ معادل یک و یک هشتم،) میتوان خروجی بخش قبل را با سه بار شیفت به راست خودش جمع کرد.

زیرماژول gender_based_multiplier_unit (مرحله دوم: اعمال ضریب جنسیت G ×)

این زیرماژول خروجی مرحله قبل (Value1) را دریافت کرده و ضریب جنسیت G را بر آن اعمال می‌کند تا Value2=Value1×G محاسبه شود. این زیرماژول به دلیل استفاده از شیفت رجیستر مبتنی بر فلیپ‌فلاپ، یک واحد **ترتیبی** است و عملکرد آن به سیگنال‌های clk_control، reset_control و enable_control وابسته است.

- **نحوه پیاده‌سازی ضریب ۱.۱۲۵ (برای زنان):**

۱. مقدار ورودی ۱۰ بیتی (value_input) ابتدا به ۱۱ بیت گسترش داده می‌شود (extended_val_comb = {1'b0, value_input}) تا از سرریز در محاسبات بعدی جلوگیری شود.

۲. برای محاسبه value/8 (بخش کسری ضریب ۱.۱۲۵)، مقدار extended_val_comb به ماژول load_and_shift_register_11bit داده می‌شود.

  - **load_and_shift_register_11bit**: این ماژول یک شیفت رجیستر ۱۱ بیتی است که با استفاده از ۱۱ نمونه از ماژول d_flip_flop_cell ساخته شده است. ابتدا مقدار ورودی (data_parallel_in) به صورت ترکیبی ۳ بیت به راست شیفت داده می‌شود (shifted_value_comb). سپس، اگر سیگنال load_en فعال باشد، این مقدار شیفت داده شده در لبه فعال کلاک (clk_main) در رجیسترهای داخلی (shifted_value_reg)

بارگذاری می‌شود. اگر load_en غیرفعال باشد، رجیسترها مقدار قبلی خود را حفظ می‌کنند. خروجی این ماژول (data_shifted_out) مقدار رجیستر شده و شیفت داده شده است.

۳. مقدار گسترش یافته اولیه (extended_val_comb – مسیر ترکیبی) و مقدار شیفت داده شده و رجیستر شده (shifted_val_reg – مسیر رجیستر شده) با استفاده از ماژول جمع‌کننده تماماً ساختاری ripple_carry_adder_11bit با هم جمع می‌شوند.

- **ripple_carry_adder_11bit**: این ماژول یک جمع‌کننده ۱۱ بیتی موجی است که با استفاده از ۱۱ نمونه از ماژول full_adder_unit ساخته شده است.

- **انتخاب خروجی و رجیستر نهایی:**
  - بر اساس سیگنال gender_input، یکی از دو مقدار (مقدار اولیه گسترش یافته برای مردان، یا حاصل جمع برای زنان) انتخاب می‌شود.
  - برای اطمینان از اینکه خروجی کل ماژول (value_output) یک مقدار پایدار و همزمان شده است، نتیجه انتخاب شده در یک رجیستر ۱۱ بیتی دیگر (result_output_reg) تحت کنترل همان سیگنال‌های کلاک، ریست و فعال‌ساز ذخیره شده و سپس به خروجی ارسال می‌گردد. این کار باعث می‌شود که خروجی این ماژول یک تأخیر حداقل یک سیکل کلاک نسبت به ورودی‌هایش داشته باشد (به علاوه تأخیر شیفت رجیستر).

- برای محاسبه بخش سوم ($\times \frac{1}{MET}$) نیز با توجه به اینکه MET چهار مقدار ممکن می تواند داشته باشد (1,2,4,8) که همه توان های ۲ هستند، می‌توان از مالتی پلکسر 4×1 وانتخاب شیفت به راست های مختلف از خروجی قبل استفاده کرد.

زیرماژول met_based_divider_unit (مرحله سوم: اعمال ضریب شدت فعالیت $\times \frac{1}{MET}$)

این زیرماژول خروجی (رجیستر شده) مرحله قبل (Value2) را دریافت کرده و آن را به صورت **ترکیبی** بر MET تقسیم می‌کند تا مقدار نهایی $T = \frac{Value2}{MET}$ به دست آید.

- **نحوه پیاده‌سازی:** از آنجایی که مقادیر MET(۱، ۲، ۴، ۸) همگی توان‌هایی از ۲ هستند، تقسیم بر MET با استفاده از عملگر شیفت به راست (>>) در Verilog پیاده‌سازی شده است. این عملگر در اینجا به عنوان یک روش ساختاری برای پیاده‌سازی تقسیم بر توان ۲ قابل قبول است، زیرا محدودیت اصلی پروژه بر عدم استفاده از عملگرهای ضرب (*) و تقسیم (/) کلی متمرکز بود.

- یک مالتی‌پلکسر ۴ به ۱ به صورت ضمنی با ساختار case (یا معادل آن با عبارت شرطی :? در کد) بر اساس مقدار met_param(۲ بیت) میزان شیفت مناسب (۰، ۱، ۲ یا ۳ بیت به راست) را برای value_input انتخاب می‌کند.
- **اشباع خروجی:** خروجی ۱۱ بیتی پس از شیفت (divided_val_comb) بررسی می‌شود. اگر این مقدار از ۲۵۵ (ماکزیمم مقدار یک عدد ۸ بیتی) بیشتر باشد، خروجی نهایی ۸ بیتی (value_output) به ۲۵۵ محدود (اشباع) می‌شود. در غیر این صورت، ۸ بیت کم ارزش divided_val_comb به عنوان خروجی انتخاب می‌گردد.

۲. واحد زمان‌بندی (ماشین حالت(FSM)) (Exercise Session FSM) : مدیریت زمان‌بندی تمرین‌ها و استراحت‌ها

این واحد، که با نام exercise_timer_fsm پیاده‌سازی شده، وظیفه کنترل توالی و زمان‌بندی دوره‌های تمرین و استراحت را بر عهده دارد. طبق تعریف پروژه، هر تمرین شامل ۴۵ ثانیه فعالیت ورزشی و به دنبال آن ۱۵ ثانیه استراحت است. این FSM تعداد کل تمرین‌ها را از واحد محاسبه دریافت کرده و برنامه تمرینی را مدیریت می‌کند.

ورودی‌ها و خروجی‌های FSM

- **ورودی‌ها:**
  - main_clk: کلاک اصلی سیستم (به عنوان مثال، کلاک برد FPGA)؛ برای تغییر حالت‌های FSM و عملیات همزمان‌سازی استفاده می‌شود.
  - one_hz_clk: کلاک با فرکانس ۱ هرتز. این کلاک برای شمارش ثانیه‌ها در تایمرهای مربوط به زمان تمرین و استراحت ضروری است.
  - reset_in: سیگنال ریست (فعال بالا در طراحی داخلی ماژول). با فعال شدن این سیگنال، FSM به حالت اولیه (STATE_IDLE) بازمی‌گردد و تمام شمارنده‌ها و رجیسترهای داخلی به مقادیر پیش‌فرض خود تنظیم می‌شوند.
  - start_signal: سیگنال ورودی از کاربر (معمولاً یک دکمه فشاری) برای شروع برنامه تمرینی از حالت STATE_IDLE یا راه‌اندازی مجدد از حالت STATE_FINAL_BEEP.

- skip_signal: سیگنال ورودی از کاربر برای رد کردن مرحله فعلی (چه تمرین و چه استراحت) و رفتن به مرحله بعدی.
- total_workout_sessions (۸ بیت): تعداد کل تمرین‌های یک دقیقه‌ای که باید انجام شود. این مقدار توسط time_calculation_logic محاسبه و به FSM ارائه می‌شود.

- **خروجی‌ها:**

- current_session_display (۸ بیت): شماره تمرین فعلی را برای نمایش به کاربر (مثلاً روی سون سگمنت) ارائه می‌دهد. در حالت STATE_FINAL_BEEP یا STATE_FINAL_BEEP معمولاً مقدار صفر یا یک کد خاص نمایش داده می‌شود.
- time_left_display (۶ بیت): زمان باقی‌مانده از مرحله فعلی (تمرین یا استراحت) را بر حسب ثانیه نمایش می‌دهد.
- activate_buzzer (۱ بیت): سیگنالی برای فعال‌سازی بازر. این سیگنال در انتهای هر مرحله که شامل تمرین و استراحت است برای مدت کوتاهی (یک سیکل one_hz_clk) و در انتهای کل برنامه تمرینی برای مدت طولانی‌تری فعال می‌شود.
- all_sessions_complete (۱ بیت): این سیگنال زمانی فعال می‌شود که تمام تمرین‌های برنامه‌ریزی شده (total_workout_sessions) با موفقیت به پایان رسیده باشند.
- session_counter_internal (۸ بیت): رجیستر داخلی برای نگهداری و شمارش شماره تمرین فعلی.

پارامترها، حالت‌ها و رجیسترهای داخلی FSM

- **پارامترهای زمانی (مدت زمان‌ها):**

- EXERCISE_TIME_SEC = 6'd45: مدت زمان هر دوره تمرین (۴۵ ثانیه).
- REST_TIME_SEC = 6'd15: مدت زمان هر دوره استراحت (۱۵ ثانیه).
- END_BEEP_TIME_SEC = 2'd2 : مدت زمان فعال بودن بازر (بر حسب تعداد سیکل one_hz_clk) پس از اتمام کل برنامه تمرینی (۲ ثانیه).

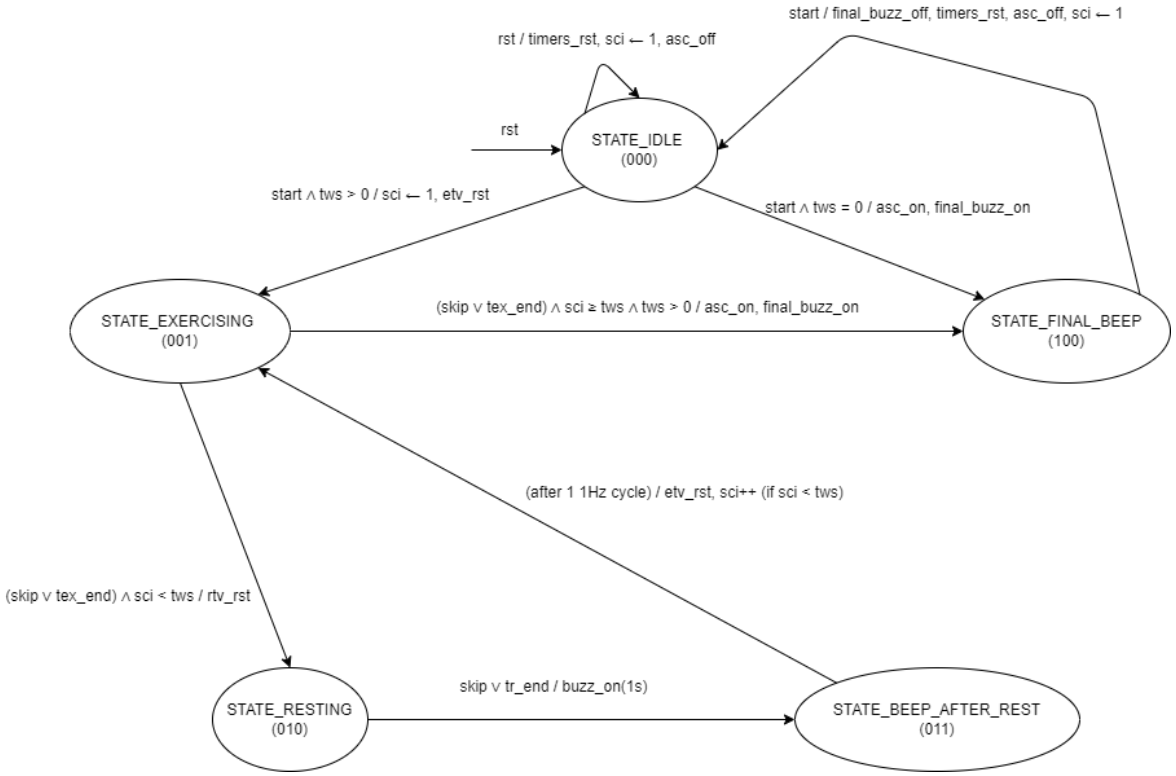
- **حالت‌های FSM (تعریف شده با پارامتر):**

- STATE_IDLE (3'b000): حالت اولیه و انتظار. سیستم منتظر دستور start_signal از کاربر است.

- (3'b001) STATE_EXERCISING: حالت انجام تمرین. تایمر exercise_timer_val در این حالت شمارش معکوس می‌کند.
- (3'b010) STATE_RESTING: حالت استراحت بین تمرین‌ها. تایمر rest_timer_val در این حالت شمارش معکوس می‌کند.
- (3'b011) STATE_BEEP_AFTER_REST: حالت کوتاه فعال‌سازی بازر پس از اتمام زمان استراحت و قبل از انتقال به STATE_EXERCISING بعدی.
- (3'b100) STATE_FINAL_BEEP: اتمام کل برنامه تمرینی و در نتیجه حالت فعال سازی بازر با فرکانسی متفاوت ازموارد قبل و انتقال به حالت اولیه و انتظار.

- **رجیسترهای حالت:**
  - current_state_reg (۳ بیت): وضعیت (حالت) فعلی FSM را نگهداری می‌کند.
  - next_state_reg (۳ بیت): وضعیت (حالت) بعدی FSM را که در منطق ترکیبی محاسبه می‌شود، نگهداری می‌کند.
- **تایمرها و شمارنده‌های داخلی:**
  - exercise_timer_val (۶ بیت): شمارنده کاهشی برای زمان تمرین .(EXERCISE_TIME_SEC) این تایمر در حالت STATE_EXERCISINGبا هر پالس کلاک یک هرتز (one_hz_clk) کاهش می‌یابد.
  - rest_timer_val(۴ بیت): شمارنده کاهشی برای زمان استراحت .(REST_TIME_SEC) این تایمر در حالت STATE_RESTING با هر پالس کلاک یک هرتز کاهش می‌یابد.
  - final_beep_timer_val (۲ بیت): شمارنده برای کنترل مدت زمان بوق در حالت) STATE_FINAL_BEEPکه حالت اتمام کل برنامه تمرینی است .(این تایمر مدت زمان فعال بودن بازر پایانی (END_BEEP_TIME_SEC) را اندازه‌گیری می‌کند.
  - session_counter_internal (۸ بیت): شماره تمرین (جلسه) فعلی را از ۱ تا total_workout_sessions می‌شمارد. این شمارنده در حالت STATE_IDLE با مقدار ۱ اولیه شده و پس از اتمام هر دوره استراحت و قبل از شروع تمرین بعدی (در حالت STATE_BEEP_AFTER_REST) افزایش می‌یابد.

تشریح عملکرد و انتقال بین حالت‌ها



راهنمای علائم اختصاری نمودار**FSM**:

- **rst**: reset_in
- **start**: start_signal
- **skip**: skip_signal
- **tex_end**: exercise_timer_val == 0
- **tr_end**: rest_timer_val == 0
- **tws**: total_workout_sessions
- **sci**: session_counter_internal
- **etv_rst**: exercise_timer_val <= EXERCISE_TIME_SEC
- **rtv_rst**: rest_timer_val <= REST_TIME_SEC

- **buzz_on(1s)**: activate_buzzer (for one one_hz_clk cycle)
- **final_buzz_on**: activate_buzzer (for END_BEEP_TIME_SEC duration)
- **final_buzz_off**: activate_buzzer (deactivated after final beep)
- **asc_on**: all_sessions_complete <= 1'b1
- **asc_off**: all_sessions_complete <= 1'b0
- **timers_rst**: Reset all timers and counters to initial values

- **STATE_IDLE (3'b000)(حالت انتظار):**
  - **نقش:** این حالت نقطه شروع سیستم پس از ریست و همچنین حالت انتظار پس از اتمام یک برنامه کامل تمرینی است. سیستم در این حالت منتظر دستور کاربر برای شروع یک برنامه جدید می‌ماند.
  - **عملیات در ورود/ماندن:** با ورود به این حالت (چه از طریق ریست یا از STATE_FINAL_BEEP)، تمام تایمرها به مقادیر اولیه خود تنظیم می‌شوند. all_sessions_complete غیرفعال شده و session_counter_internal برای شروع از جلسه اول به 8'd1 مقداردهی می‌شود. خروجی‌های نمایشی معمولاً مقادیر صفر یا پیش‌فرض را نشان می‌دهند.
  - **انتقال:** با دریافت سیگنال start_signalاز کاربر:
    - اگر total_workout_sessions(تعداد کل جلسات محاسبه شده توسط واحد محاسبه) بزرگتر از صفر باشد، FSM به حالت STATE_EXERCISING منتقل می‌شود تا اولین جلسه تمرین آغاز شود. تایمر تمرین برای این جلسه ریست می‌شود.
    - اگر total_workout_sessions صفر باشد (یعنی بر اساس ورودی‌های کاربر، هیچ تمرینی لازم نیست)، FSM مستقیماً به STATE_FINAL_BEEP منتقل می‌شود تا این وضعیت به کاربر اعلام گردد.
- **STATE_EXERCISING (3'b001)(حالت انجام تمرین):**
  - **نقش:** این حالت مسئول مدیریت زمان ۴۵ ثانیه‌ای فعالیت ورزشی است.
  - **عملیات در ورود/ماندن** : تایمر exercise_timer_val با هر پالس one_hz_clk یک واحد کاهش می‌یابد. شماره جلسه فعلی (session_counter_internal) و زمان باقی‌مانده تمرین (exercise_timer_val) برای نمایش به کاربر از طریق خروجی‌های current_session_display و time_left_display آماده می‌شوند.
  - **انتقال:** اگر سیگنال skip_signal از کاربر دریافت شود (به معنی تمایل کاربر برای رد کردن ادامه تمرین فعلی) یا اگر exercise_timer_val به صفر برسد (زمان تمرین به طور طبیعی تمام شود):
    - بررسی می‌شود آیا جلسه فعلی، آخرین جلسه برنامه‌ریزی شده بوده است یا خیر (با مقایسه session_counter_internalو total_workout_sessions).

- اگر آخرین جلسه بوده باشد (و total_workout_sessions > 0تا از حالت بدون تمرین متمایز شود)، FSMبه حالت STATE_FINAL_BEEPمنتقل می‌شود تا پایان کل برنامه تمرینی اعلام شود.
- در غیر این صورت (هنوز جلسات تمرینی باقی مانده است)، FSM به حالت STATE_RESTINGمنتقل می‌شود تا دوره استراحت ۱۵ ثانیه‌ای آغاز گردد. در این مدل ۵ حالته، بوقی بلافاصله پس از اتمام زمان تمرین و قبل از شروع استراحت در نظر گرفته نشده است. تایمر استراحت برای این دوره ریست می‌شود.

- **STATE_RESTING (3'b010)(حالت استراحت):**
  - **نقش:** این حالت مسئول مدیریت زمان ۱۵ ثانیه‌ای استراحت بین جلسات تمرین است.
  - **عملیات در ورود/ماندن:** تایمر rest_timer_val با هر پالس one_hz_clk یک واحد کاهش می‌یابد. شماره جلسه فعلی (که در واقع جلسه تمرینی است که به تازگی تمام شده یا جلسه بعدی که قرار است شروع شود) و زمان باقی‌مانده استراحت برای نمایش آماده می‌شوند.
  - **انتقال:** اگر سیگنال skip_signalدریافت شود یا اگر rest_timer_val به صفر برسد (زمان استراحت تمام شود)، FSM به حالت STATE_BEEP_AFTER_REST منتقل می‌شود.
- **STATE_BEEP_AFTER_REST (3'b011)(حالت بوق پس از استراحت):**
  - **نقش:** این حالت برای ایجاد یک بوق کوتاه صوتی طراحی شده تا کاربر را از پایان دوره استراحت و شروع قریب‌الوقوع جلسه تمرین بعدی مطلع سازد. این حالت تنها برای یک سیکل کلاک one_hz_clk فعال است.
  - **عملیات در ورود/ماندن:** سیگنال activate_buzzer برای یک سیکل one_hz_clk فعال می‌شود. تایمر تمرین (exercise_timer_val) برای جلسه بعدی به مقدار اولیه (EXERCISE_TIME_SEC)ریست می‌شود. شمارنده جلسه (session_counter_internal) یک واحد افزایش می‌یابد تا برای جلسه تمرین بعدی آماده شود (این افزایش تنها در صورتی رخ می‌دهد که جلسه فعلی آخرین جلسه نبوده باشد، هرچند با توجه به منطق انتقال از STATE_EXERCISING، اگر آخرین جلسه بود، سیستم اصلاً وارد STATE_RESTING و سپس این حالت نمی‌شد).
  - **انتقال (پس از یک سیکل one_hz_clk) FSM :**به طور خودکار به حالت STATE_EXERCISING منتقل می‌شود تا جلسه تمرین بعدی آغاز گردد.

- **STATE_FINAL_BEEP (3'b100) (حالت اتمام کل برنامه تمرینی):**
    - **نقش:** این حالت نشان‌دهنده پایان موفقیت‌آمیز کل برنامه تمرینی (یا عدم نیاز به تمرین در ابتدا) است. یک بوق متمایز (طولانی‌تر یا با فرکانس متفاوت، که در اینجا با مدت زمان END_BEEP_TIME_SEC کنترل می‌شود) برای اعلام این وضعیت به کاربر تولید می‌شود.
    - **عملیات در ورود/ماندن:** سیگنال all_sessions_completeفعال (یک منطقی) می‌شود. سیگنال activate_buzzerبرای مدت زمان END_BEEP_TIME_SECثانیه (که توسط شمارنده final_beep_timer_valو کلاک one_hz_clkکنترل می‌شود) فعال می‌ماند. خروجی‌های نمایشی مقادیر پایانی (مثلاً صفر یا یک کد خاص برای حالت اتمام) را نشان می‌دهند.
    - **انتقال:** با دریافت سیگنال start_signalاز کاربر، سیستم به حالت STATE_IDLEبازمی‌گردد و برای شروع یک برنامه تمرینی جدید آماده می‌شود. در این انتقال، شمارنده‌ها و سیگنال all_sessions_completeریست می‌شوند.

ماژول سطح بالا (exercise_system_top_level)

این ماژول (exercise_systemtop_level) نمونه‌سازی و اتصال ماژول‌های time_calculation_logic و exercise_session_fsm را انجام می‌دهد. همچنین ورودی/خروجی‌های کلی سیستم را مدیریت می‌کند:

- سیگنال ریست فعال پایین (reset_button_n) از برد را به ریست فعال بالا برای ماژول‌های داخلی تبدیل می‌کند.
- ورودی‌های کاربر از DIP سوئیچ‌ها (user_dip_switches) را به ورودی‌های time_calculation_logic نگاشت می‌دهد.
- خروجی تعداد کل تمرین‌ها از time_calculation_logic را به ورودی total_workout_sessions در exercise_session_fsm متصل می‌کند.
- سیگنال‌های کنترلی و کلاک را به ماژول‌ها ارسال می‌کند.
- خروجی‌های FSM (مانند بازر و LED اتمام) را به پین‌های خروجی متصل می‌کند.
- خروجی‌های نمایشی FSM برای اتصال به درایور سون سگمنت (که در فازهای بعدی پیاده‌سازی می‌شود) آماده هستند و فعلاً به مقادیر پیش‌فرض تنظیم شده‌اند.

نتیجه‌گیری فاز اول

در فاز اول، معماری سیستم زمان‌بندی تمرین با موفقیت طراحی و در Verilog توصیف شد. واحد محاسبه با استفاده از LUT و عملیات ساختاری، تعداد تمرین‌ها را محاسبه می‌کند. ماشین حالت ۵ وضعیتی نیز جریان تمرین و استراحت را با دقت کنترل می‌نماید. این طراحی، پایه محکمی برای مراحل شبیه‌سازی و پیاده‌سازی سخت‌افزاری در فازهای بعدی پروژه فراهم می‌کند.