



THE IMAGINATION UNIVERSITY PROGRAMME

RVfpga Deney 8

Zamanlayıcılar

1. GİRİŞ

Donanım zamanlayıcıları mikrodenetleyicilerle yongadaki sistemlerde bulunan yaygın çevre birimleridir. Genelde kesin zamanlama oluşturmak için kullanılırlar. Zamanlayıcılar bir sayacı sabit bir sıklıkta, ki bu sıklık genelde yapılandırılabilir, artırır ya da azaltır, ardından sayaç sıfıra ya da önden tanımlanmış bir değere gelince işlemciye kesinti yapar. Daha gelişmiş zamanlayıcılar bir motorun hızını ya da bir ışığın parlaklığını denetlemek için darbe-genişlik modülasyonlu (PWM) dalga biçimleri oluşturma gibi başka işlevleri de gerçekleştirir.

Bu deneyde, önceki deneyler gibi bir yapı kullanarak, ilk RVfpga'de içerilen zamanlayıcının yüksek-düzeyle standardını tanımlayıp, ardından alçak-düzeyle gerçekleştirmesini açıklıyoruz. Bir zamanlayıcının nasıl kullanılacağını ya da değiştireceğini gösteren temel, gelişmiş alıştırmalar sunuyoruz.

2. RVfpga'de İÇERİLEN ZAMANLAYICININ YÜKSEK DÜZEYLİ SPESİFİKASYONU

Bu bölümde RVfpga'de kullanılan zamanlayıcının yüksek düzeyli spesifikasyonunu çözümleyip ardından bu çevre birimini kullanan bir alıştırma öneriyoruz.

A. Zamanlayıcı yüksek düzeyli spesifikasyonu

RVfpga'de kullanılan zamanlayıcı modülü OpenCores'dan alınmıştır (<https://opencores.org/projects/ptc>). Paketi indirirsen modülün yüksek düzeyli spesifikasyonunu tanımlayan bir belge içerisinde sağlar (ki bunu biz de burada sağlıyoruz: [RVfpgaPath]/RVfpga/src/SweRVolfSoC/Peripherals/ptc/docs/ptc_spec.pdf). Zamanlayıcının ana işlemiyle özelliklerini burada özetliyoruz; ancak eksiksiz bilgi yukarıdaki belgede bulunabilir.

Zamanlayıcı modülünün şu ana özellikleri vardır:

- Bir Wishbone Ara Bağlantısı kullanır
- 32-bit sayaç/zamanlayıcı tesisi
- PWM/Zamanlayıcı/Sayacın (PTC) bir kez ya da sürekli çalıştırması
- Programlanabilir PWM (Darbe-genişlik modülasyonlu) modu
- Zamanlayıcı işlevi için sistem saati ile dış saat kaynakları
- HI/LO Referans, Yakalama yazmaçları
- PWM çıktı sürücüsü için üç-durum denetlemesi
- PTC işlevleri CPU'ya kesinti verebilir

Zamanlayıcı modül spesifikasyonu dokümanının Bölüm 4'ü zamanlayıcı modülü içerisindeki değişik adreslere atanmış denetleme, durum yazmaçlarını tanımlar (Tablo 1'e göz at). RVfpga'de zamanlayıcının taban adresi **0x80001200**'dir.

Tablo 1. Zamanlayıcı Yazmaçları

Adı	Adresi	Width	Erişim	Tanım
RPTC_CNTR	0x80001200	1-32	R/W	Main PTC counter
RPTC_HRC	0x80001204	1-32	R/W	PTC HI Reference/Capture register
RPTC_LRC	0x80001208	1-32	R/W	PTC LO Reference/Capture register
RPTC_CTRL	0x8000120C	9	R/W	Control register

RPTC_CNTR yazmacı asıl sayaç yazmacıdır, bütün sayaç/zamanlayıcı saat dönümlerinde artırılır. RPTC_CTRL yazmacı zamanlayıcı modülünü denetlemek için kullanılır; Tablo 2 bitlerinin işlevini gösterir. RPTC_HRC ile RPTC_LRC referans/yakalama yazmacı olarak kullanılır.

Tablo 2. RPTC_CTRL bitleri

Bit	Erişim	Reset	Ad & Tanım
0	R/W	0	EN When set, RPTC_CNTR increments.
1	R/W	0	ECLK Selects the clock signal: external clock, through <i>ptc_ecgt</i> (1), or system clock (0).
2	R/W	0	NEC Used for selecting the negative/positive edge and low/high period of the external clock (<i>ptc_ecgt</i>).
3	R/W	0	OE Enables PWM output driver.
4	R/W	0	SINGLE When set, RPTC_CNTR is not incremented after it reaches value equal to the RPTC_LRC value. When cleared, RPTC_CNTR is restarted after it reaches value in the RPTC_LCR register.
5	R/W	0	INTE When set, PTC asserts an interrupt when RPTC_CNTR value is equal to the value of RPTC_LRC or RPTC_HRC. When the signal is cleared, interrupts are masked.
6	R/W	0	INT When read, this bit represents pending interrupt. When it is set, an interrupt is pending. When this bit is written with '1', interrupt request is cleared.
7	R/W	0	CNTRRST When set, RPTC_CNTR is reset. When cleared, normal operation of the counter occurs.
8	R/W	0	CAPTE When set, RPTC_CNTR is captured into RPTC_LRC or RPTC_HRC registers. When cleared, capture function is masked.

GÖREV: Zamanlayıcı modülünde RPTC_CNTR, RPTC_HRC, RPTC_LRC, RPTC_CTRL yazmaçlarının bildirimini, bunların adreslerinin tanımlarını (sırasıyla 0x80001200, 0x80001204, 0x80001208, 0x8000120C) bul. Zamanlayıcı modülüne şu klasörden erişilebilir [*RVfpgaPath*]/*RVfpga/src/SweRVolfSoC/Peripherals/ptc*.

Zamanlayıcı değişik modlarda işlem görebilir (bu deneyde kullanacağın modları kısaca tanımlıyoruz; daha çok detay için zamanlayıcı modül spesifikasyon dokümanının Bölüm 3'üne bak):

- **Zamanlayıcı/Sayaç modu:** Bu modda sistem saati ya da dış saat referansı eğer sayaç etkinse (RPTC_CTRL[EN]=1) RPTC_CNTR yazmacını artırır. RPTC_CNTR, RPTC_LRC'ye eşitken RPTC_CTRL[INTE] ayarlıysa RPTC_CTRL[INT] yükseğe gider.
- **PWM modu:** Darbe Genişlik Modülasyonu (PWM) bir Sinyal, dijital bir kaynaktan analog bir sinyal oluşturmak için bir yöntemdir. Bir PWM sinyali davranışını tanımlayan iki değerden oluşur: *görev dönümü* ile *sıklık*. Görev dönümü, bir dönümü bitirmek için geçen toplam sürenin yüzdesi olarak sinyalin yüksekte olduğu süreyi tanımlar. Sıklık ise dönümün ne sıklıkta tekrarladığıdır. Bir dijital sinyali yeterince hızlı bir oranda

kapatıp açınca çıktı, aygıtlara güç sağlarken sabit voltajlı bir analog sinyal gibi davranır görünecektir. Örneğin 50% görev dönümlü (dönümün yarısında yüksekte) bir sinyal ile 3.3 V'lık yüksek voltaj bir analog yüklemeye 1.67 V olarak görünecektir (dönüm süresince ortalama voltaj). Aynı sinyal 33% görev dönümüyle 1.1V olarak görünür. PWM modunda işlem yapmak için RPTC_CTRL[OE] ayarlı olmalı. RPTC_HRC ile RPTC_LRC yazmaçları PWM çıktı sinyalinin yüksek ile düşük periyotlarına ayarlı olmalıdır: PWM sinyali, sıfırlamanın (RPTC_CNTR'da) RPTC_HRC saat dönümü sonrasında yükseğe gitmeli; sıfırlamanın (RPTC_CNTR'da) RPTC_LRC saat dönümü sonrasında düşüğe gitmeli.

3. TEMEL ALIŞTIRMA

Alıştırma 1. 8 sayı 7 kesimli ekranlarda artan bir sayı gösteren bir program yaz. Değer saniyede bir değişmeli, bu gecikmeyi oluşturmak için zamanlayıcı modülü kullanılmalıdır.

- İlk olarak programı RISC-V çevirici dilinde yazıp Nexys A7 kartında çalıştır.
- Ardından aynı programla Verilator'da simülasyon gerçekleştir. Şu sinyalleri ekleyebilirsin: sistem saati, 8-sayı 7-kesimli ekranlarda gösterilecek değeri depolayan işlemci yazmacı, RPTC_CNTR, RPTC_LRC, RPTC_HRC, RPTC_CTRL zamanlayıcı yazmaçları.
- Şimdi programı C'de yazıp Nexys A7 kartında çalıştır.
- C programının Verilator'da aşama (b) içerisinde RISC-V çevirici programına yaptığın gibi simülasyonunu yap.

4. ZAMANLAYICININ ALÇAK DÜZEYLİ GERÇEKLEŞTİRMESİ

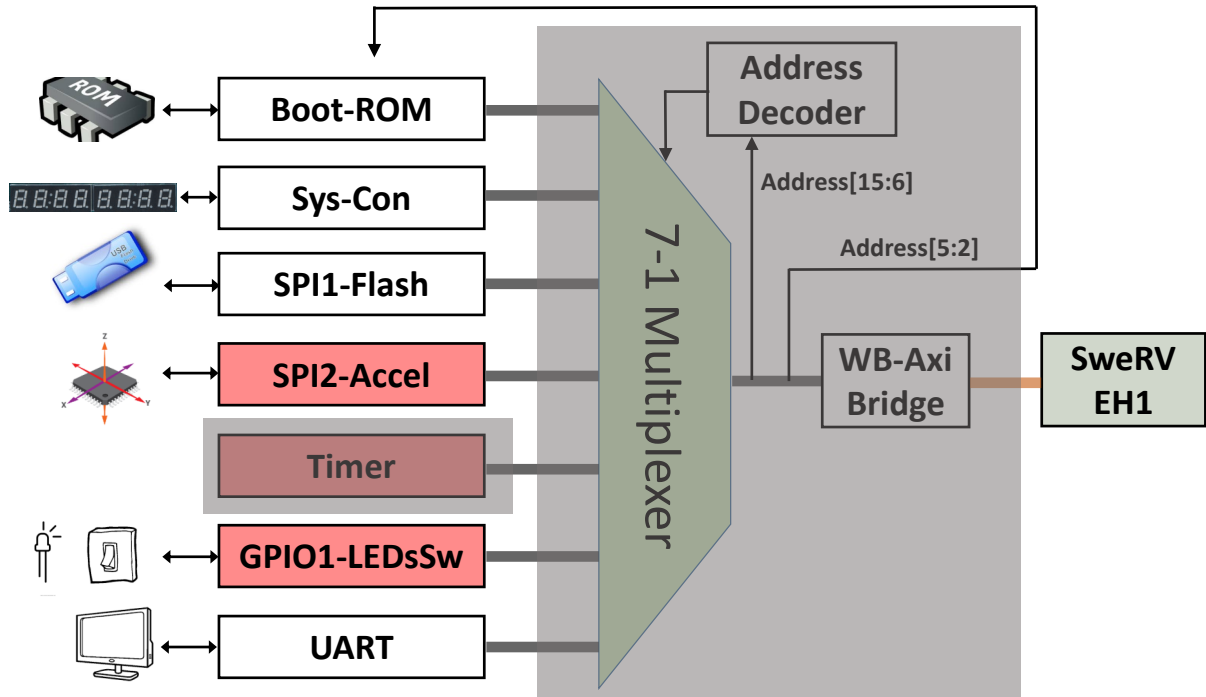
Bu bölümde ilk olarak RVfpga'deki zamanlayıcı modülünün alçak düzeyli gerçekleştirmesini tanımlayıp ardından ilk modülü değiştirip ardından Nexys A7 kartındaki üç renkli LEDleri denetlemek için programda kullanacağın birkaç alıştırmayı öneriyoruz.

A. Zamanlayıcının alçak düzeyli gerçekleştirmesi

Önceki deneylerde izlediğimiz şemadaki gibi zamanlayıcı modülünün çözümlemesini fazlara bölüyoruz.

- Yeni modülün RVfpga'de entegrasyonu (Figür 1'de sol taralı bölge)
- Yeni modül ile SweRV EH1 Çekirdeği arasındaki bağlantı (Figür 1'de sağ taralı bölge).

Önemli olarak, önceki deneylerle karşılaştırılınca, bu çevre birimi (zamanlayıcı) Nexys A7 kartına fiziksel olarak bağlı değil. Zamanlayıcı RVfpga'in içerisindedir.



Figür 1. 2 fazda zamanlayıcı modül çözümlemesi

i. Zamanlayıcı modülünün SoC'de entegrasyonu

swervolf_core ([RVfpgaPath]/RVfpga/src/SweRVolfSoC/swervolf_core.v) modülünün 361-379 arası satırlarında zamanlayıcı modülünün somutlaması yapılır (Figür 2'ye göz at).

```

358 // PTC
359 wire      ptc_irq;
360
361 ptc_top timer_ptc(
362     .wb_clk_i    (clk),
363     .wb_rst_i    (wb_rst),
364     .wb_cyc_i    (wb_m2s_ptc_cyc),
365     .wb_adr_i    ({2'b0,wb_m2s_ptc_adr[5:2],2'b0}),
366     .wb_dat_i    (wb_m2s_ptc_dat),
367     .wb_sel_i    (4'b1111),
368     .wb_we_i     (wb_m2s_ptc_we),
369     .wb_stb_i    (wb_m2s_ptc_stb),
370     .wb_dat_o    (wb_s2m_ptc_dat),
371     .wb_ack_o    (wb_s2m_ptc_ack),
372     .wb_err_o    (wb_s2m_ptc_err),
373     .wb_inta_o   (ptc_irq),
374     // External PTC Interface
375     .gate_clk_pad_i (),
376     .capt_pad_i (),
377     .pwm_pad_o (),
378     .oen_padoen_o ()
379 );

```

Figür 2. Zamanlayıcı modülün entegrasyonu (dosya swervolf_core.v).

Modülün arayüzü iki bloğa bölünebilir: Wishbone sinyalleri (Tablo 3), Dış I/O sinyalleri (Tablo 4). Wishbone sinyalleri, SweRV EH1 Çekirdeğinin zamanlayıcıyla bir denetleyici/çevre birimi modeliyle iletişim kurmasını sağlar. Dış I/O sinyalleri zamanlayıcı modülünü dış aygıtlarla bağlar; örneğin, *pwm_pad_o* yukarıda tanımlanan PWM modunda işlem yaparken PWM çıktı

sinyali sağlar (zamanlayıcı modüllerini üç renkli LEDlerle bağlamak için Alıştırma 2’de bu sinyali kullanman gerekecek).

Tablo 3. Wishbone Sinyalleri

Port	Genişlik	Yön	Tanım
wb_cyc_i	1	Inputs	Indicates valid bus cycle (core select)
wb_adr_i	15	Inputs	Address inputs
wb_dat_i	32	Inputs	Data inputs
wb_dat_o	32	Outputs	Data outputs
wb_sel_i	4	Inputs	Indicates valid bytes on data bus (during valid cycle, this signal must be 0xf)
wb_ack_o	1	Output	Acknowledgment output (indicates normal transaction termination)
wb_err_o	1	Output	Error acknowledgment output (indicates an abnormal transaction termination)
wb_rty_o	1	Output	Not used
wb_we_i	1	Input	Write transaction when asserted high
wb_stb_i	1	Input	Indicates valid data transfer cycle
wb_inta_o	1	Output	Interrupt output

Tablo 4. Dış I/O Sinyalleri

Port	Genişlik	Yön	Tanım
gate_clk_pad_i	1	Input	External clock / Gate input
capt_pad_i	1	Input	Capture input
pwm_pad_o	1	Output	PWM output
oen_padoen_o	1	Output	PWM output driver enable (for three-state or open-drain driver)

Figür 2’nin satır 365’inde gösterildiği gibi, Wishbone veri yolu sinyalinde (*wb_m2s_ptc_adr[5:2]*) çekirdekten sağlanan adresin bits [5:2]’si 4 erişilebilir yazmaç arasından seçmek için kullanılır (Bellek Eşlenmiş I/O). Dolayısıyla, RPTC_CNTR yazmacına 0x80001200 adresinde, RPTC_HRC yazmacına 0x80001204 adresinde, RPTC_LRC yazmacına 0x80001208 adresinde, RPTC_CTRL yazmacına 0x8000120C adresinde erişebiliriz.

ii. Zamanlayıcı ile SweRV EH1 Çekirdeği arasındaki bağlantı

Önceki deneylerde açıklandığı gibi, aygıt denetleyicileri SweRV EH1 Çekirdeğine bir Çoklayıcı üzerinden bağlıdır (Figür 1). 7:1 çoklayıcının (Figür 3) şu dosyada gerçekleştirildiğini anımsa

[RVfpgaPath]/RVfpga/src/SweRVolfSoC/Interconnect/WishboneInterconnect/wb_intercon.v,
ki şu dosyanın 104-205 arası satırlarında somutlaması yapılır

[RVfpgaPath]/RVfpga/src/SweRVolfSoC/Interconnect/WishboneInterconnect/wb_intercon.vh.
İkinci dosya şuradaki **swervolf_core** modülünün satır 168’inde içerilir:

[RVfpgaPath]/RVfpga/src/SweRVolfSoC/swervolf_core.v.

```

108 wb_mux
109 #(.num_slaves (7),
110 .MATCH_ADDR ({32'h00000000, 32'h00001000, 32'h00001040, 32'h00001100, 32'h00001200, 32'h00001400, 32'h00002000}),
111 .MATCH_MASK ({32'hffffff00, 32'hffffffc0, 32'hffffffc0, 32'hffffffc0, 32'hffffffc0, 32'hffffffc0, 32'hffffff00}))
112 wb_mux_io
113 (.wb_clk_i (wb_clk_i),
114 .wb_rst_i (wb_rst_i),
115 .wbm_adr_i (wb_io_adr_i),
116 .wbm_dat_i (wb_io_dat_i),
117 .wbm_sel_i (wb_io_sel_i),
118 .wbm_we_i (wb_io_we_i),
119 .wbm_cyc_i (wb_io_cyc_i),
120 .wbm_stb_i (wb_io_stb_i),
121 .wbm_cti_i (wb_io_cti_i),
122 .wbm_bte_i (wb_io_bte_i),
123 .wbm_dat_o (wb_io_dat_o),
124 .wbm_ack_o (wb_io_ack_o),
125 .wbm_err_o (wb_io_err_o),
126 .wbm_rty_o (wb_io_rty_o),
127 .wbs_adr_o (wb_rom_adr_o, wb_sys_adr_o, wb_spi_flash_adr_o, wb_spi_accel_adr_o, wb_ptc_adr_o, wb_gpio_adr_o, wb_uart_adr_o),
128 .wbs_dat_o (wb_rom_dat_o, wb_sys_dat_o, wb_spi_flash_dat_o, wb_spi_accel_dat_o, wb_ptc_dat_o, wb_gpio_dat_o, wb_uart_dat_o),
129 .wbs_sel_o (wb_rom_sel_o, wb_sys_sel_o, wb_spi_flash_sel_o, wb_spi_accel_sel_o, wb_ptc_sel_o, wb_gpio_sel_o, wb_uart_sel_o),
130 .wbs_we_o (wb_rom_we_o, wb_sys_we_o, wb_spi_flash_we_o, wb_spi_accel_we_o, wb_ptc_we_o, wb_gpio_we_o, wb_uart_we_o),
131 .wbs_cyc_o (wb_rom_cyc_o, wb_sys_cyc_o, wb_spi_flash_cyc_o, wb_spi_accel_cyc_o, wb_ptc_cyc_o, wb_gpio_cyc_o, wb_uart_cyc_o),
132 .wbs_stb_o (wb_rom_stb_o, wb_sys_stb_o, wb_spi_flash_stb_o, wb_spi_accel_stb_o, wb_ptc_stb_o, wb_gpio_stb_o, wb_uart_stb_o),
133 .wbs_cti_o (wb_rom_cti_o, wb_sys_cti_o, wb_spi_flash_cti_o, wb_spi_accel_cti_o, wb_ptc_cti_o, wb_gpio_cti_o, wb_uart_cti_o),
134 .wbs_bte_o (wb_rom_bte_o, wb_sys_bte_o, wb_spi_flash_bte_o, wb_spi_accel_bte_o, wb_ptc_bte_o, wb_gpio_bte_o, wb_uart_bte_o),
135 .wbs_dat_i (wb_rom_dat_i, wb_sys_dat_i, wb_spi_flash_dat_i, wb_spi_accel_dat_i, wb_ptc_dat_i, wb_gpio_dat_i, wb_uart_dat_i),
136 .wbs_ack_i (wb_rom_ack_i, wb_sys_ack_i, wb_spi_flash_ack_i, wb_spi_accel_ack_i, wb_ptc_ack_i, wb_gpio_ack_i, wb_uart_ack_i),
137 .wbs_err_i (wb_rom_err_i, wb_sys_err_i, wb_spi_flash_err_i, wb_spi_accel_err_i, wb_ptc_err_i, wb_gpio_err_i, wb_uart_err_i),
138 .wbs_rty_i (wb_rom_rty_i, wb_sys_rty_i, wb_spi_flash_rty_i, wb_spi_accel_rty_i, wb_ptc_rty_i, wb_gpio_rty_i, wb_uart_rty_i));
139
140 endmodule

```

CPU/Controller Signals

Peripheral Signals

Figür 3. CPU'ya bağlı çevre birimini seçen 7-1 çoklayıcı (dosya *wb_intercon.v*)

Çoklayıcı okunacak ya da yazılacak çevre birimini seçer, böylelikle CPU'yu (*wb_io_** sinyalleri – 115-126 arası satırlar, Figür 3) bir çevre biriminin Wishbone Veri Yoluna (127-138 arası satırlar, Figür 3), adresle değişecek biçimde (110-111 arası satırlar), bağlar. Örneğin, CPU'nun oluşturduğu adres 0x80001200-0x8000123F aralığında ise zamanlayıcı modülü seçilir, dolayısıyla *wb_io_** sinyalleri *wb_ptc_** sinyalleriyle bağlanır.

5. İLERİ DÜZEY ALIŞTIRMALAR

Alıştırma 2. Zamanlayıcının PWM çıktı sinyalini (*pwm_pad_o*) Nexys A7 kartında erişilebilir iki üç renkli LEDlerden birine bağlamak için RVfpga'i değiştir. Bu yeni yetkinliği Deney 6 ile 7'de değiştirdiğin güncellenmiş RVfpga sistemine eklemen önerilir.

- Digilent, Nexys A7 kartında erişilebilir üç renkli LEDler üzerine şu bilgileri sağlar: <https://reference.digilentinc.com/reference/programmable-logic/nexys-a7/reference-manual>
- Yukarıdaki belgeyi özetlemek gerekirse, kart, iki üç renkli LED barındırır. Bütün üç renkli LEDlerin içerisinde üç küçük LED için üç ayrı girdi sinyali güdümlü katot vardır: **kırmızı** için, **mavi** için, **yeşil** için. Bunlardan birini yükseğe sürmek karşılık gelen iç LED'i aydınlatacaktır. Üç renkli LED, aydınlatılan iç LEDlerin kombinasyonuyla bağlı bir renk çıkaracaktır. Örneğin kırmızıyla maviyi yükseğe sürmek mor bir renk çıkarır. Diligent üç renkli LEDleri sürerken PWM kullanımını önerir. Bu girdilerden birini düz mantık '1'e sürmek LED'in gereksiz parlaklıkta aydınlatılmasına neden olacaktır. Üç renkli sinyallerin kesinlikle 50%'den yüksek olmayan görev dönümüyle sürüldüğünün sağlamasını yaparak bunu engelleyebilirsin. Dahası, bir PWM kullanımı üç renkli LED'in potansiyel renk paletini de çok büyük oranda büyötmektedir. Renklerin görev dönümünü ayrı ayrı olarak 50% ile 0% arasında ayarlamak değişik renklerin değişik yoğunlukta aydınlatılmasına neden olur, böylelikle de neredeyse bütün renkler gösterilebilir.
- RVfpga'de bulunan zamanlayıcıyı taban olarak üç yeni zamanlayıcı modülü oluştur. Bütün renkler (kırmızı, mavi, yeşil) değişik zamanlayıcı modülü güdümlü olmalıdır, ki ayrı voltajlar alabilsinler.
- Yeni zamanlayıcı başına yazmaçları belleğe eşlemek için şu adresleri kullan:
 - Zamanlayıcı-2: 0x80001240-0x8000127F
 - Zamanlayıcı-3: 0x80001280-0x800012BF
 - Zamanlayıcı-4: 0x800012C0-0x800012FF

Önemli olarak bu durumda çevre birimini seçen çoklayıcıya 3 yeni girdi eklemen gerekir (Figür 1).

- 3 rengin şu kart uçlarına bağlandığını göz önünde bulundurarak kısıtlandırma dosyasını değiştirmen gerek:
 - iv. LED16_B \leftrightarrow PIN R12
 - v. LED16_G \leftrightarrow PIN M16
 - vi. LED16_R \leftrightarrow PIN N15

Alıştırma 3. 16 anahtarın sağladığı değerle üç renkli LEDi denetlemek için yeni çevre birimini kullanan bir program gerçekleştir. Mavi rengin görev dönümünü ayarlamak için en sağdaki 5 anahtarı kullan, sonraki 5 anahtarı yeşil için, sonraki 5 anahtarı ise kırmızı için. (En soldaki anahtar kullanılmayacak.)

- a. İlk olarak programı RISC-V çeviricisinde yaz.
- b. Ardından programı C'de yaz.