گزارش کار سوال دوم تمرین سری چهارم سیستم‌های عامل – محمدپویا تراشی – ۴۰۱۵۲۱۱۰۲

تابع Sender:

```
 1. void* sender(void* arg) {
 2.     int sender_id = *(int*)arg;
 3.     while (1) {
 4.         Packet pkt;
 5.         pkt.id = rand() % 1000;
 6.         snprintf(pkt.message, sizeof(pkt.message), "Message from sender %d", sender_id);
 7.         pkt.pid = sender_id;
 8.         pkt.ack = 0;
 9.
10.         pthread_mutex_lock(&cb.lock);
11.
12.         if (cb.count == BUFFER_SIZE)
13.         {
14.             printf("Buffer full. Dropping packet with id %d\n", pkt.id);
15.         }
16.         else
17.         {
18.             cb.buffer[cb.tail] = pkt;
19.             cb.tail = (cb.tail + 1) % BUFFER_SIZE;
20.             cb.count++;
21.             printf("Sender %d sent packet with id %d\n", sender_id, pkt.id);
22.         }
23.
24.         pthread_mutex_unlock(&cb.lock);
25.
26.         struct timespec timeout;
27.         clock_gettime(CLOCK_REALTIME, &timeout);
28.         timeout.tv_sec += ACK_TIMEOUT;
29.
30.         pthread_mutex_lock(&lock_conds[sender_id]);
31.         int ret = pthread_cond_timedwait(&ack_conds[sender_id], &lock_conds[sender_id],
&timeout);
32.
33.         if (ret == ETIMEDOUT)
34.             printf("Timeout happened. No consumer sent ACK for packet %d sent from Sender
%d\n", pkt.id, pkt.pid);
35.         else
36.             printf("Sender %d received ACK for packet with id %d\n", sender_id, pkt.id);
37.
38.         pthread_mutex_unlock(&lock_conds[sender_id]);
39.
40.         sleep(rand() % 3);
41.     }
42. }
43.
```

در ابتدا محتویات پکت ساخته می‌شود (خطوط ۴-۸). سپس mutex مربوط به بافر قفل می‌شود و در صورتی‌که بافر پر نباشد، نخ ارسال‌کننده وارد بخش ارسال packet می‌شود و در نهایت mutex بافر را باز می‌کند. سپس با بررسی زمان سیستم و افزودن محدودهٔ زمانی تایم‌اوت به آن، زمان منقضی شدن packet به دست می‌آید. با استفاده از تابع pthread_cond_timedwait، شرط ack_conds مربوط به پکت ارسالی خود را بررسی می‌نماید. در صورتی‌که یک مصرف‌کننده این پکت را دریافت کند یا زمان مورد نظر تمام شود، این تابع از حالت wait بیرون می‌آید. در صورتی‌که خروجی آن صفر باشد پکت توسط یک نویسنده دریافت شده است. در صورتی‌که مقدار بازگشتی این تابع برابر ETIMEDOUT باشد، هیچ مصرف‌کننده‌ای پکت را دریافت نکرده است.

تابع Consumer:

```
 1. void* consumer(void* arg) {
 2.     while (1) {
 3.         pthread_mutex_lock(&cb.lock);
 4.
 5.         if (cb.count != 0)
 6.         {
 7.
 8.             Packet pkt = cb.buffer[cb.head];
 9.             cb.head = (cb.head + 1) % BUFFER_SIZE;
10.             cb.count--;
11.
12.             printf("Consumer consumed packet with id %d, message: %s\n", pkt.id,
pkt.message);
13.             pkt.ack = 1;
14.             printf("Consumer sending ACK for packet id %d to sender %d\n", pkt.id, pkt.pid);
15.
16.             pthread_cond_signal(&ack_conds[pkt.pid]);
17.         }
18.
19.         pthread_mutex_unlock(&cb.lock);
20.
21.         sleep(rand() % 3);
22.     }
23. }
24.
```

در ابتدای این تابع قفل مربوط به بافر بسته می‌شود تا فرستنده یا گیرندهٔ دیگری نتواند تغییراتی روی بافر اعمال کند. در صورت خالی نبودن بافر، پکتی که head به آن اشاره می‌کند مصرف می‌شود ACK مربوط به آن ۱ شده و سیگنالی را روی شرط مربوط به پکت ارسال می‌کند تا timedwaitای که نویسنده در حال انجام آن است، در صورتی‌که تا الان به علت زمان خاتمه نیافته، پایان یابد.