Министерство образования Республики Беларусь Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №4 по дисциплине ОСиСП

«Задача производители-потребители для процессов»

Выполнил: студент группы 150501 Климович А.Н. Проверил: старший преподаватель _____ Поденок Л.П.

1 ПОСТАНОВКА ЗАДАЧИ

Основной процесс создает очередь сообщений, после чего ожидает и обрабатывает нажатия клавиш, порождая и завершая процессы двух типов — производители и потребители.

Очередь сообщений представляет собой классическую структуру — кольцевой буфер, содержащий указатели на сообщения, и пара указателей на голову и хвост. Помимо этого очередь содержит счетчик добавленных сообщений и счетчик извлеченных.

Производители формируют сообщения и, если в очереди есть место, перемещают их туда.

Потребители, если в очереди есть сообщения, извлекают их оттуда, обрабатывают и освобождают память с ними связанную.

Для работы используются два семафора для заполнения и извлечения, а также мьютекс или одноместный семафор для монопольного доступа к очереди.

Сообщения имеют следующий поля: размер (size), тип (type), хэш (hash) и сами данные (data).

Производители генерируют сообщения, используя системный генератор rand(3) для size и data. В качестве результата для size используется остаток от деления на 257.

Если остаток от деления равен нулю, rand(3) вызывается повторно. Если остаток от деления равен 256, значение size устанавливается равным 0, реальная длина сообщения при этом составляет 256 байт.

При формировании сообщения контрольные данные формируются из всех байт сообщения. Значение поля hash при вычислении контрольных данных принимается равным нулю. Для расчета контрольных данных можно использовать любой подходящий алгоритм на выбор студента.

После помещения значения в очередь перед освобождением мьютекса очереди производитель инкрементирует счетчик добавленных сообщений. Затем после поднятия семафора выводит строку на stdout, содержащую помимо всего новое значение этого счетчика.

Потребитель, получив доступ к очереди, извлекает сообщение и удаляет его из очереди. Перед освобождением мьютекса очереди инкрементирует счетчик извлеченных сообщений. Затем семафора после поднятия проверяет stdout, контрольные данные И выводит строку на

содержащую помимо всего новое значение счетчика извлеченных сообщений.

При получении сигнала о завершении процесс должен завершить свой цикл и только после этого завершиться, не входя в новый.

Следует предусмотреть задержки, чтобы вывод можно было успеть прочитать в процессе работы программы.

Следует предусмотреть защиту от тупиковых ситуаций из-за отсутствия производителей или потребителей.

2 АЛГОРИТМ

2.1 Main

- 1) Начало.
- 2) Выделить общую разделяемую память для кольцевого буфера сообщений.
 - 3) Инициализация флагов кольцевого буфера.
 - 4) Создание семафоров.
 - 5) Инициализация семафоров.
 - 6) Бесконечный цикл.
 - 7) Ввод опции.
 - 8) Выполнение выбранной опции.
 - 9) Конец бесконечного цикла.
 - 10) Удаление общей памяти.
 - 11) Удаление семафоров.
 - 12) Конец.

2.2 Producer

- 1) Начало.
- 2) Получение доступа к семафорам.
- 3) Подключение к общей памяти.
- 4) Бесконечный цикл.
- 5) Генерация сообщения.
- 6) Конец бесконечного цикла.
- 7) Отсоединение от общей памяти.
- 8) Конец.

2.3 Consumer

- 1) Начало.
- 2) Получение доступа к семафорам.
- 3) Подключение к общей памяти.
- 4) Бесконечный цикл.

- 5) Извлечение сообщений из буфера.
- 6) Конец бесконечного цикла.
- 7) Отсоединение от общей памяти.
- 8) Конец.

155 26 216 145 248 192 151 66 242

3 ТЕСТИРОВАНИЕ ПРОГРАММЫ

```
$ ./main
Enter 'h' for help
+p -- add producer
+c -- add consumer
-p -- delete producer
-c -- delete consumer
l -- list of processes
q -- exit
<<<<<<<<<<<<<<<<<<
Producer_1 has been created, pid=6248
Consumer_1 has been created, pid=6249
236 170 249 178 211 116 135 117 243 64 2 51 82 163 54 138 213 90 52
12 247 153 211 97 84 194 209 207 156 107 144 136 148 7 185 102 124
191 219 238 255 92 32 208 127 215 89 211 48 13 223 167 166 178 7 121
243 216 199 142 66 87 21 86 222 77
217 11 22 70 137 114 102 89 112 188 49 67 108 190 33 18 99 82 153 221
68 112 163 211 49 121 103 135 215 181 195 175 63 88 245 201 73 219 161 186 21 81 253 15 157 146 242 240 42 77 51 25 112 133 75 233 237
81 62 32 147 109 223 235 225 38 52 58 199 109 80 152 105 208 167 133
98 152 116 140 230 39 166 213 43 112 60 151 64 123 183 212 232 21 61
200 59 113 130 2 93 210 25 69 161 192 203 130 215 62 14 59 229 51 15
16 34 76 167 98 199 222 53 45 243 243 116 45 99 246 175 193 70 200 5
103 6 79 233 222 13 118 152 242 40 168 129 74 115 168 173 185 133 97
230 247 83 217 163 54 78 82 118 19 153 251 250 31 74 227 124 87 216 19 200 58 73 202 173 241 118 101 245 87 202 235 170 34
95 112 94 213 2 119 79 253 150 153 94 145 111 53 164 54 181 223 255
127 10 239 116 112 98 203 185 76 117 219 89 83 202 54 40 205 173 119
72 194 143 166 82 254 219 118 179 15 212 178 142 222 31 1 205 1 205
134 205 193 224 37 19 41 220 187 117 7 177 189 202 63 98 155 188 188
144 239 203 99 31 216 193 190 218 141 191 37 146 140 230 114 48 120
155 11 179 15 19 99 75 92 34 173 118 93 232 6 75 179 233 106 9 169
168 227 54 102 135 200 242 236 185 34 100 211 173 22 97 63 249 172
```

```
List of processes:
_____
Main process: pid = 6246
_____
PRODUCERS[all = 1]:
Producer_1, pid=6248
_____
CONSUMERS[all = 1]:
Consumers_1, pid=6249
<<<<<<<<<<<<<<<<<<
16 216 206 230 127 22 2 223 13 105 230 163 35 239 106 63 12 132 102 2
139 145 182 16 124 101 171 74 130 212 130 17 42 207 119 170 101 121
136 114 98 110 149 4 220 126 67 103 130 40 105 12 186 30 156 53
70 127 132 25 129 21 195 79 140 108 180 4 116 166 230 97 58 234 60 55
45 163 185 213 139 68 142 41 224 196 172 37 194 176 190 66 197 129 17
80 108 68 212 224 105 186 192 34 34 252 90 79 29 18 164 169 87 49 210
54 116 253 220 54 43 153 248 240 153 136 191 5 205 147 100 53 204 36
88 238 159 178 189 60 196 96 229 154 145 53 209 133 177 172 187 221
196 49 204 221 186 139 226 134 157 197 58 104 104 146 213 135 195 145
195 6 241 38 160 1 91 112 134 11 155 192 232 223 241 180 187
190 27 46 90 224 105 65 200 122 21 78 189 38 144 195 22 182 98 151 16
82 156 155 108 219 2 74 75 53 4 116 243 31 163 204 127 139 12 198 4
+p
194 227 82 5 252 185 247 117 152 16 160 219 232 115 252 67 204 221
160 36 14 97 218 148 43 220 168 162 141 59 159 206 157 241 83 23 40 73 140 63 90 172 26 65 30 149 3 106 242 34 142 256 132 231 148 46 194
187 208 206 117 111 156 17 223 110 41 135 54 52 198 144 95 224 80 126
117 212 232 230 117 245 229 120 219 248 166 157 179 118 106 167 100
133 185 66 243 97 201 41 149 14 56 245 238 8 242 226 91 88 199 208 76
43 72 38 34 109 66 84 227 44 252 198 177 51
Producer_2 has been created, pid=6250
35 148 208 204 168 222 131 27 204 139 140 44 101 228 115 181 175 29
124 85 191 233 151 19 75 195 142 16 244 193 152 22 212 104 226 251 69
100 22 144 111 162 60 83 5 175 7 180 75 131 136 9 108 31 156 54 97 41
70 84 105 223 234 188 198 203 183 10 175 205 26 157 238 86 240 243 132 119 38 207 121 174 216 229 76 116 154 45 28 225 129 134 62 107 65
3 181 119 141 227 67 167 127 177 124 239 163 256 101 72 77 222 118 37
66 65 24 220 110 180 59 240 185 121 218 122 124 142 241 9 241 180 47
111 100 172
============End of eating=====================
```

-c
Consumer_1 has been killed
BUFFER IS FULL
BUFFER IS FULL
BUFFER IS FULL
BUFFER IS FULL
-pBUFFER IS FULL

Producer_2 has been killed BUFFER IS FULL BUFFER IS FULL BUFFER IS FULL BUFFER IS FULL -pBUFFER IS FULL

Producer_1 has been killed q All producers are killed All consumers are killed

4 ВЫВОД

Были разработаны три программы: main, producer, consumer. Программа main порождает и удаляет процессы производителей producer и потребителей consumer. Все программы обмениваются информацией, используя общую память, а также синхронизируют свою работу с этой памятью при помощи бинарных семафоров.