

Xilinx Zynq FPGA, TI DSP MCU 기반의

프로그래밍 및 회로 설계
전문가

강사 이상훈
(Innova Lee)

Gcccompil3r@gmail.com

학생 김민호

minking12@naver.com

```

1 #include<unistd.h>
2 #include<stdio.h>
3 #include<errno.h>
4 #include<stdlib.h>
5 #include<fcntl.h>
6 #include<sys/types.h>
7 #include<sys/wait.h>
8 #include<sys/stat.h>
9 #include<signal.h>
10
11 /*
12 void term_status(int status)
13 {
14     if(WIFEXITED(status)) // 정상 종료
15         printf("(exit)status : 0x%x\n", WEXITSTATUS(status));
16     else if(WTERMSIG(status)) // 비정상 종료
17         printf("(signal)status : 0x%x, %s\n", status & 0x7f,
18             WCOREDUMP(status) ? "core dumped" : "");
19 }
20
21 int main(void)
22 {
23     pid_t pid;
24     int status;
25     if((pid = fork()) > 0)
26     {
27         wait(&status); // 자식 프로세스 작업이 끝날 때 까지 대기, 자식 프로세스가 종료한 상태를 구함
28         term_status(status);
29     }
30     else if(pid == 0)
31         abort(); // abort 는 signal 을 보낸 것이다. 6번
32     else
33     {
34         perror("fork() ");
35         exit(-1);
36     }
37     return 0;
38 }
39 */

```

```

41 /*
42 void term_status(int status)
43 {
44     if(WIFEXITED(status))//정상종료
45         printf("(exit)status : 0x%x\n", WEXITSTATUS(status));
46     else if(WTERMSIG(status))//비정상종료
47         printf("(signal)status : 0x%x, %s\n",
48             status & 0x7f, WCOREDUMP(status) ? "core dumped" : "");
49 }
50 }
51
52 void my_sig(int signo) // sigchild 전달
53 {
54     int status;
55     wait(&status);
56     term_status(status);
57 }
58
59 int main()
60 {
61     pid_t pid;
62     int i;
63     signal(SIGCHLD, my_sig); // sigchld 가 나올 때 my_sig 를 실행하라.자식 프로세스가 죽으면 무조건 실행 ...
64     if((pid = fork()) > 0)
65         for(i = 0; i < 1000; i++)
66         {
67             usleep(50000); // 0.05초 간격으로 조건문 실행
68             printf("%d\n", i + 1);
69         }
70     else if(pid == 0)
71         sleep(5); // 5초후에 my_sig 함수 실행
72     else
73     {
74         perror("fork() ");
75         exit(-1);
76     }
77     return 0;
78 }
79

```

```

83
84 int main()
85 {
86     execlp("ps", "ps", "-e", "-f", 0); //첫 ps는 명령어, 그 이후엔 인자(인덱스) 0은 널 값
87     return 0;
88 }
89 */
90
91 /*
92 int main()
93 {
94     execlp("ps", "ps", "-e", "-f", 0); //둔갑술 ? 형태가 바뀐다. 메모리에 있는 형태가 바뀐다. 메모리를 ps 로 바꾼다. 메모리 레이아웃이 ps로 변경
95     printf("after\n"); //실행안됨.
96     return 0;
97 }
98 */
99

```

```
100 /*
101 int main()
102 {
103     int status;
104     pid_t pid;
105     if((pid=fork())>0)
106     {
107         wait(&status);//자식 프로세스 우선 실행 이랑 같다.
108         printf("prompt>\n");//ps -e -f 기능 실행후 문자열 출력.
109     }
110     else if(pid==0)
111     {
112         execlp("ps","ps","-e","-f",0);
113     }
114     return 0;
115 }
116 */
```

```

118 /*
119 //newpgm.c
120 int main(int argc,char **argv)
121 {
122     int i;
123     for(i=0;argv[i];i++)
124         printf("argv[%d]=[%s]\n",i,argv[i]);
125     return 0;
126 }
127 */
128
129 /*
130 int main()
131 {
132     int status;
133     pid_t pid;
134     if((pid=fork())>0)
135     {
136         wait(&status);
137         printf("prompt>\n");
138     }
139     else if(pid==0)
140     {
141         execl("./newpgm","newpgm","one","two",(char *)0);//내가 만든 newpgm 을 쓴다는것. newpgm,one two를 받아서 출력함,캐릭터 포인터로 널을 변환.
142     }
143     return 0;
144 }
145 */
146

```

```

168 /
169 int main()
170 {
171     system("date");//system은 내부적으로 fork 를 한후에 exec 를 한 것.
172     printf("after\n");
173     return 0;
174 }
175 */

```

```

177
178 /*
179 int my_system(char *cmd)
180 {
181     pid_t pid;
182     int status;
183     char *argv[]={"sh","-c",cmd,0};//shell, -c : 해당 커맨드 실행 , 입력으로 들어온 cmd= date실행
184     char *envp[]={0};
185     if((pid=fork())>0)
186         wait(&status);
187     else if(pid==0)
188
189
190         execve("/bin/sh",argv,envp);//다른 프로그램 실행 =bin/sh -c date
191
192 }
193
194 int main()
195 {
196     for(;;)
197     {
198         my_system("date");
199         sleep(1);//1초 간격으로 ...
200     }
201     printf("after\n");
202     return 0;
203 }
204 */
205

```