

# **Xilinx Zynq FPGA, TI DSP, MCU**

## 기반의 프로그래밍 및 회로 설계

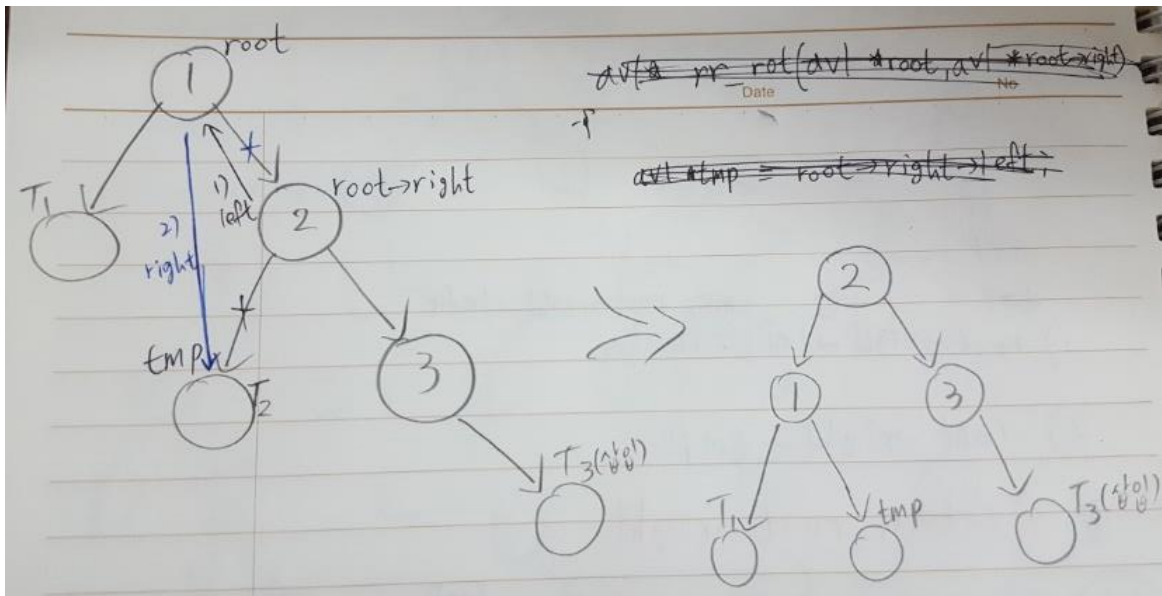
### 전문가 과정

**강사 – Innova Lee(이상훈)**  
[gcccompil3r@gmail.com](mailto:gcccompil3r@gmail.com)

**학생 – 안상재**  
[sangjae2015@naver.com](mailto:sangjae2015@naver.com)

## 1) RR 회전

- 노드가 오른쪽으로 일직선상에 있고, 오른쪽 높이가 왼쪽 높이보다 2 이상 클 때 수행함.



- 내가 구현해본 RR 회전 코드

```

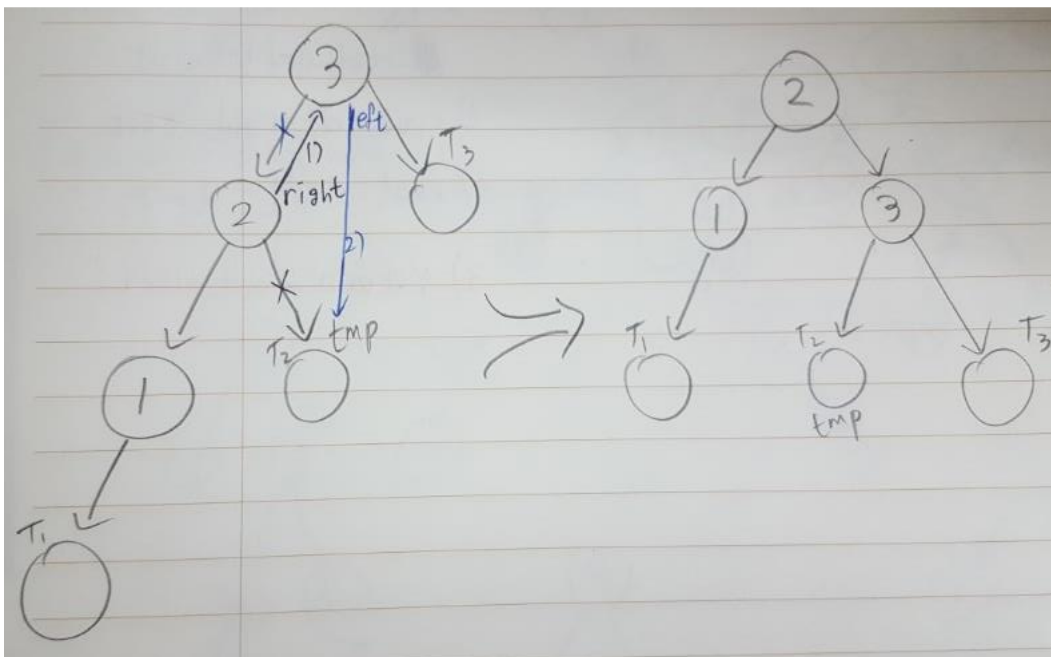
avl * rr_rot(avl * root, avl * root->right)    // 현재 root는 1이고 (*root)->right는 2
{
    avl * tmp = root->right->left;           // 2 노드의 left가 1 노드를 가리키면 2와 T2 노드가 끊어지므로
                                              T2에 접근할 수 없어짐. 그래서 T2를 tmp에 잠시 저장함.
    root->right->left = root;                // 2 노드의 left가 1노드를 가리키게함.
    root->right = tmp;                      // 1 노드의 right가 2 노드의 left였던 T2를 가리키게 함.
    return root->right;                    // 가장 위의 노드인 2의 노드를 리턴함.
}

```

- \* 핵심 : 2의 왼쪽 노드를 어떻게 처리할 것인가?

## 2) LL 회전

- 노드가 왼쪽으로 일직선상에 있고, 왼쪽 높이가 오른쪽 높이보다 2 이상 클 때 수행함.



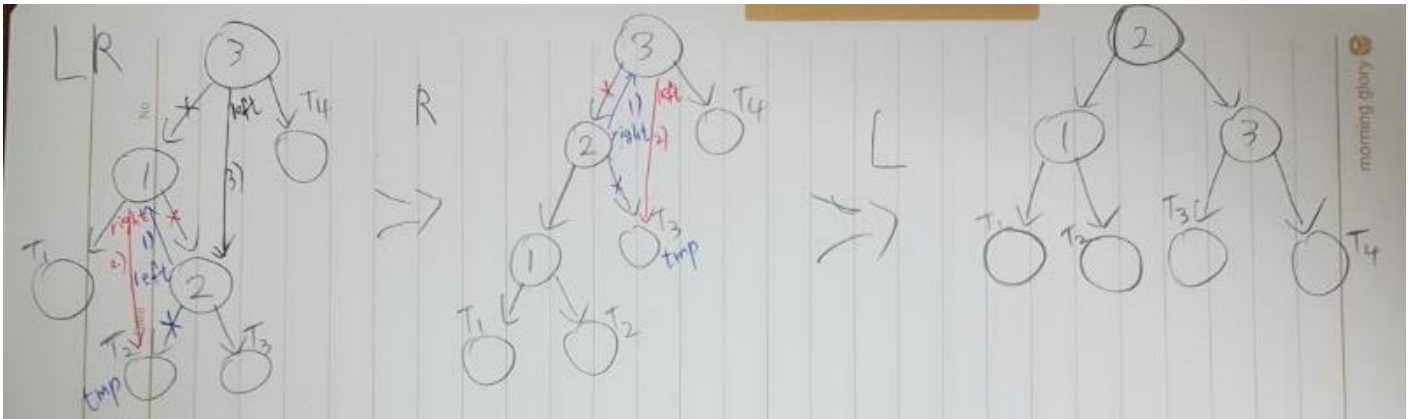
- 내가 구현해본 LL 회전 코드 (RR과 동일한 방법)

```
avl* ll_rot(avl *root, avl *root->left)
{
    avl *tmp = root->left->right;
    root->left->right = root;
    root->left = tmp;
    return root->left;
}
```

\* 핵심 : 2의 오른쪽 노드를 어떻게 처리할 것인가?

### 3) LR 회전 코드

- LR 회전은 1 노드의 오른쪽 자식의 왼쪽에 노드가 추가될 때 수행된다.



- 내가 구현해본 LR 회전 코드

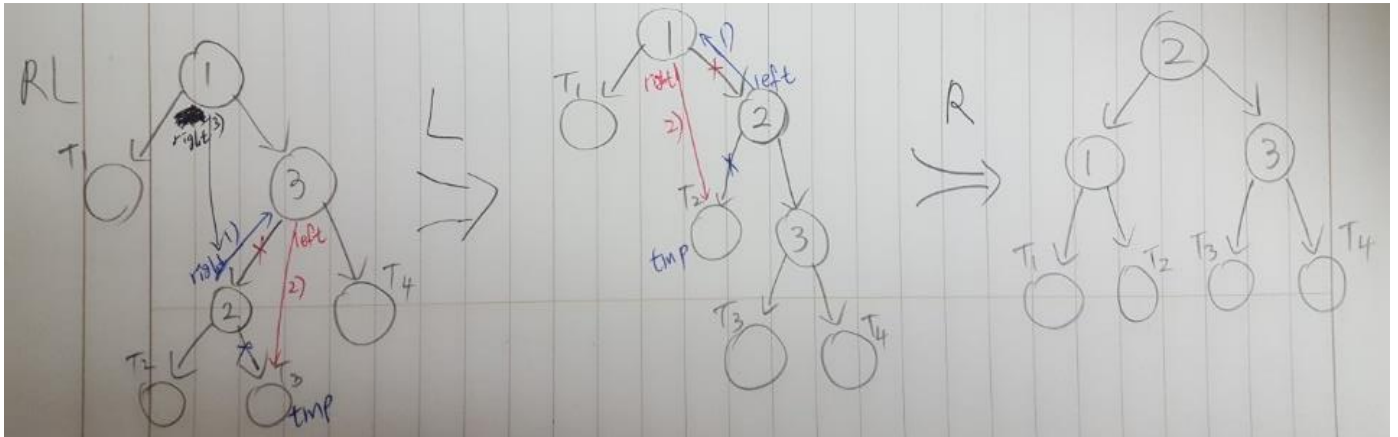
```
avl* lr_rot(avl *root, avl *root->left)
{
    avl *tmp1;          // 2 노드를 저장할 포인터 변수.
    avl *tmp = root->left->right->left; // 2 노드의 left가 1 노드를 가리키면 2와 T2 노드가 끊어지므로
                                                T2에 접근할 수 없어짐. 그래서 T2를 tmp에 저장함.
    root->left->right->left = root->left; // 2 노드의 left가 1 노드를 가리키게 함.
    root->left->right = tmp;           // 1 노드의 right가 T2 가리키게 함.
    root->left = root->left->right;    // 3 노드의 left가 2 노드를 가리키게 함.

    tmp1 = root->left;               // 3 노드의 left가 T3을 가리키면 3 노드에서 2 노드를 접근할 수 없게됨.
                                    그래서 2 노드를 tmp1에 저장함.
    tmp = root->left->right;          // 2 노드의 right가 3 노드를 가리키면 T3에 접근할 수 없게됨.
                                    그래서 T3을 tmp에 저장함.
    root->left->right = root; // 2 노드의 right가 3 노드를 가리키게 함.
    root->left = tmp;        // 3 노드의 left가 T3을 가리키게 함.

    return tmp1; // root가 된 2 노드의 주소를 리턴함.
}
```

#### 4) RL 회전 코드

- 3의 왼쪽 자식의 오른쪽 노드에 새로운 노드가 삽입되면 LR 회전이 수행된다. 1, 2에 대해 RR회전이 수행되고 그 후 3을 기준으로 LL 회전이 수행된다.



- 내가 구현해본 RL 회전 코드 (LR 회전 코드와 동일한 방법)

```

avl* rl_rot(avl *root, avl *root->right)
{
    avr *tmp1;
    avl *tmp = root->right->left->right;
    root->right->left->right = root->right;
    root->right->left = tmp;
    root->right = tmp;
    root->right = root->right->left;

    tmp1 = root->right;
    tmp = root->right->left;
    root->right->left = root;
    root->right = tmp;

    return tmp1;
}

```