

# Computer Arithmetic Project

## Names:

- HADIOUCHE. Azouaou
- SMAI. Abdelmalek

## Task 1/2: C Implementations of Radix- $2^r$ and SSP.

### Utils

To make the implementation easier, we implemented multiple functions in `utils.c`.

- `tobinary` and `tobinaryl`: two functions that take a number and return a string with the binary representation of the number, and can take a length for which they would display.
- `extract`: takes a number and returns a slice starting from `start` until `start+length`
- `bitlength`: takes a number and returns how many bits are needed to store it.
- `Word` and `Sentence`: structs that help with C strings.
- `prepend` and `append`: functions to interact with `Sentence` structs to add and remove contents from them.
- `tostring`: after the processing of a `Sentence` with `prepends` and `appends`, returns a string with the contents of added to `Sentence`.

### Radix- $2^r$

We heavily use bitwise operations like shifting and bitmasking to make the operations as fast as possible.

- `pad`: takes a number and adds a zero in the LSB.
- `rslice`: takes a number and returns the `index`-th slice starting from `window * index` to `window * (index + 1) + 1`, adds the LSB to the remaining bits and then turns it into two complementary representation.
- `factor2`: gets the power of 2 in the decomposition  $m_i 2^{n_i}$  of the slices  $Q_i$ .
- `radix-2r`: the function that returns a string of the radix- $2^r$  representation of a number given a window, for later calculations, we added a pointer to ones that calculates how many non-zeros elements are in the representation.

### SSP

Same as Radix- $2^r$ , in that case we do implement the code directly from the paper given at page 3.

- `sslice`: same as the slice from radix- $2r$ , but in the case it doesn't add the LSB to the remaining bits.
- `nextone`: given a start point and a number, it looks for the location of the next 1 in the binary representation of the number, which is needed in our case to do the skip zeros step of SSP.
- `ssp`: takes a number and a window and returns the SSP representation of the number, notice that we add some bits in the MSB of the representation, which can be discarded since they are 0s but the implementation would take time to add such a feature.

### Main

- `testbench`: shows the representation of the first 4096 numbers in binary, radix- $2r$  and `ssp`, also shows the hamming weight of each for the representations.
- `calculate_averages`: it calculates the values to display in the tables that will be below, the calculations will take lots of time especially for the case of 32 bits table.

## Task 3:

Here we present the results of the calculations of `calculate_averages` and calculations related to the smallest number that requires  $q$  additions.

In the first table, we use the formulas that were given in course and paper and that are deducible from using some combinatorics to fill the contents of the theoretical hamming weights.

Bitlength of the constant $n$	8	12	16	20	24	28	32
Bitlength of the slice $w$	3		4				
$A_\omega$ of Radix- $2^w$	2.625	4.375	4.6875	5.625	6.5625	7.5	8.4375
$EA_\omega$ of Radix- $2^w$	2.6367	4.0004	5.3750	6.1406	7.5009	8.6250	9.1241
$A_\omega$ of SSP	2.2879	3.2840	3.4081	4.2056	5.0031	5.8006	6.5981
$EA_\omega$ of SSP	2.6289	3.8205	5.2588	6.2349	7.4209	8.6200	9.0521
Saving of SSP Over Radix- $2^w$	0.2461	0.6094	0.7324	0.6395	1.0087	1.0943	1.5612

Notice that in average the hamming weight of SSP is almost equivalent to Radix- $2^r$ .

Saving of SSP Over Radix- $2^w$	7.80e-03	1.80e-01	1.16e-01	-9.43e-02	8.00e-02	5.00e-03	7.20e-02
Relative Saving of SSP Over R2r	0.30%	4.50%	2.16%	-1.54%	1.07%	0.06%	0.79%

Another measure of improvement is in the calculation of a product, in the table below, we show the first value of  $n$  that requires  $q$  additions in the calculation multiplication with 32 bit constants. This question turns back to display the first number that has a hamming weight equal to  $q - 1$  which is done in the program.

$q$	Radix- $2^r$	SSP
1	5	4
2	21	33
3	85	260
4	341	2081
5	1365	16644
6	5461	133153
7	21845	1065220
8	87381	8521761

We can notice that the Radix- $2^r$  performs a lot better than SSP in this case.