stosfr (SRAM to SFR)

Type	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	opcode = 0x05							R0						R1							-					-						-

- [R0] decoding table

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
[63:32]	sfr_address																-			fetch_limit[18:8]												
[31:0]	fetch_limit[7:0]								-		fetch_base																					

- [R1] decoding table

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
[63:32]	words_per_packet							size															data_offset									
[31:0]	dim1_log_size								dim0_log_size							slice_log_size							topology									

Field Spec:
- *fetch_base*:
    - unit = byte, alignment = 8
- *fetch_limit*:
    - unit = byte, alignment = 8
- *sfr_address*:
    - unit = byte, alignment = 1
- *topology*:
    - {direct = 0, dim0br = 1, dim1br = 2, tr_dim1br = 9}
- *slice_log_size*:
    - {6, 7, 8}
- *dim0_log_size*:
    - {0, 1, 2, 3, 4, 5, 6, 7, 8}
- *dim1_log_size*:
    - {0, 1, 2, 3, 4, 5, 6, 7, 8}
- *data_offset*:
    - {0, 1, 2, 3, 4, 5, 6, 7, 8n}, unit = byte
- *size*:
    - {1, 2, 4, 8n}, unit = byte
- *words_per_packet*:
    - {1, 2, 4n}

Requirement:

```Python
fetch_limit % (words_per_packet*8) == 0

# if size < 8
(sfr_address // 8) == {(sfr_address + size) // 8} and (data_offset < 8) and (sfr_address % 8 >=
data_offset)
```