Problem 1.

Part A. WLOG assume that $l = 1$, otherwise we just send each bit separately. For each $i \in \{1, \ldots, t\}$, the receiver chooses $b = 1$ if $i = x$ and $b = 0$ otherwise; the sender chooses $\hat{m}_0 = 0$ and $\hat{m}_1 = m_i$. Then the receiver uses the 1-out-of-2 OT to get $v_i = \hat{m}_b$. The result is $v_1 \vee \cdots \vee v_t$.

To simulate the view of sender, we just call $\mathsf{OT.Sim}_S\left((0, m_i), \perp\right)$ for $i \in \{1, \ldots, t\}$. To simulate the view of receiver, we just call $\mathsf{OT.Sim}_T(0, 0)$ for $i \neq x$ and $\mathsf{OT.Sim}_T\left(1, m_x\right)$ for $i = x$. Therefore the scheme is secure.

Part B. The protocol goes like this:

- The sender prepares t random messages $r_1, \cdots, r_t$ uniformly sampled from $\{0,1\}^\ell$.
- The given 1-out-of-2 OT protocol is used for t rounds. In round i, the sender prepares the following two inputs for the OT protocol:

$$\left(r_i, \bigoplus_{j<i} r_j \oplus m_i\right)$$

- If the receiver wants to learn m_x, he or she requires the former message in the first $x - 1$ rounds, and the later message in round x, which means he or she can learn

$$r_1, \cdots, r_{x-1}, \bigoplus_{j<x} r_j \oplus m_x$$

  helping him or her reveal m_x.

The correctness for this protocol is obvious. The view of the sender can be simulated by t $\mathsf{OT.Sim}_S$, which runs $\mathsf{OT.Sim}_S$ on t pairs of inputs prepared by the sender independently.

The view of the receiver can also be simulated by t $\mathsf{OT.Sim}_R$. In the first $x-1$ rounds, the (semi-honest) receiver can only require the former message, and t OT. Sim $_R$ simply samples $r \leftarrow \$$ and returns $\mathsf{OT.Sim}_R(r, 0)$ to the receiver. Of course it should remember all r-s. When round x comes, the receiver requires the later message, the t $\mathsf{OT.Sim}_R$ will return $\mathsf{OT.Sim}_R\left(m_x \oplus R, 1\right)$, where R denotes the XOR sum of all r-s. In the remaining rounds, it simply returns OT.$\mathsf{Sim}_R(\$, b)$ for the require bit b from the receiver. Thus, the protocol is secure against semi-honest sender and semi-honest receiver.

Problem 2.

Part A. For each $i \in \{n_{\text{in}} + 1, \ldots, n\}$, evaluate

$$L_{i,v_i} = \mathsf{Dec}\left(L_{j_2,v_{j_2}}[1], \mathsf{Dec}\left(L_{j_1,v_{j_1}}[1], c_{i,L_{j_1,v_{j_1}}[0],L_{j_2,v_{j_2}}[0]}\right)\right)$$

For each $i \in \{n - n_{\text{out}} + 1, \ldots, n\}$, output $v_i = L_{i,v_i}$ (assume that $L_{i,0} = 0$ and $L_{i,1} = 1$).

Part B. For each wire j, assume j is one of the input wires of d gates, then $L_{j,0}$ and $L_{j,1}$ will be used as key for $2d$ times. To use only one-time pad, we can generate $2d$ different keys for $L_{j,0}$ (and $L_{j,1}$), so every key will be only used once.

For a gate with output wire i among these d gates, the two keys generated for this gate should be of length $|L_{i,0}|$ (or $|L_{i,1}|$). Therefore, assume the output wire of these d gates are $i_1, \ldots, i_d$, respectively, then $|L_{j,0}| = \sum_{k=1}^{d} 2\,|L_{i_k,0}| + 1$, where $+1$ stands for the mask bit. It is easy to use induction to prove that

$$|L_{j,0}| = \sum_{d} 2^d \times [\text{ the number of paths with length } d \text{ starting from wire } j]$$

Therefore, for circuits with depth $O(\log n)$, the length of input label is $\text{poly}(n)$, so they can be efficiently garbled.

Part C. Here we prove the security of Part A. The simulator works as follows:

1. Sim samples $L_1, \cdots, L_{n_{in}}$ uniformly at random, and use them as the input labels. It also samples $L'_1[1], \cdots, L_{n_{in}}[1]'$ uniformly at random, and generate the complementary labels $L'_1, \cdots, L'_{n_{in}}$.
2. Find a gate such that both of its input labels have been determined, and its c has not been determined. Let L_i, L_j denote its input labels, and L'_i, L'_j be their complements. If it is not an output gate, then Sim samples an output label L_k uniformly at random. It also samples $L'_k[1]$, and generate a complementary output label L'_k.
3. Sim computes $c_{k,L_i[0],L_i[0]}$ using L_i, L_j, L_k. For the other three cyphertexts, it uses L_k or L'_k arbitrarily.
4. Repeat Step 2 and 3 until the c of all gates have been determined.
5. Output the generated c as $\tilde{C}$, and $(L_1, \cdots, L_{n_{in}})$ as $\left(L_{1,x_1}, \cdots, L_{n_{in},x_{n_{in}}}\right)$.

We then show that the generated view is computationally indistinguishable from the real view. First, $(L_1, \cdots, L_{n_{in}})$ is sampled uniformly. Then for all gates, the "correct" output is always consistent. Notice that all the "complementary" labels are not present in the generated view. Thus by the CPA-security of the encryption scheme, the distribution of the three "incorrect" cyphertexts in the generated view is computationally indistinguishable from the real view, which completes the proof.